

Figure 1
(Prior Art)

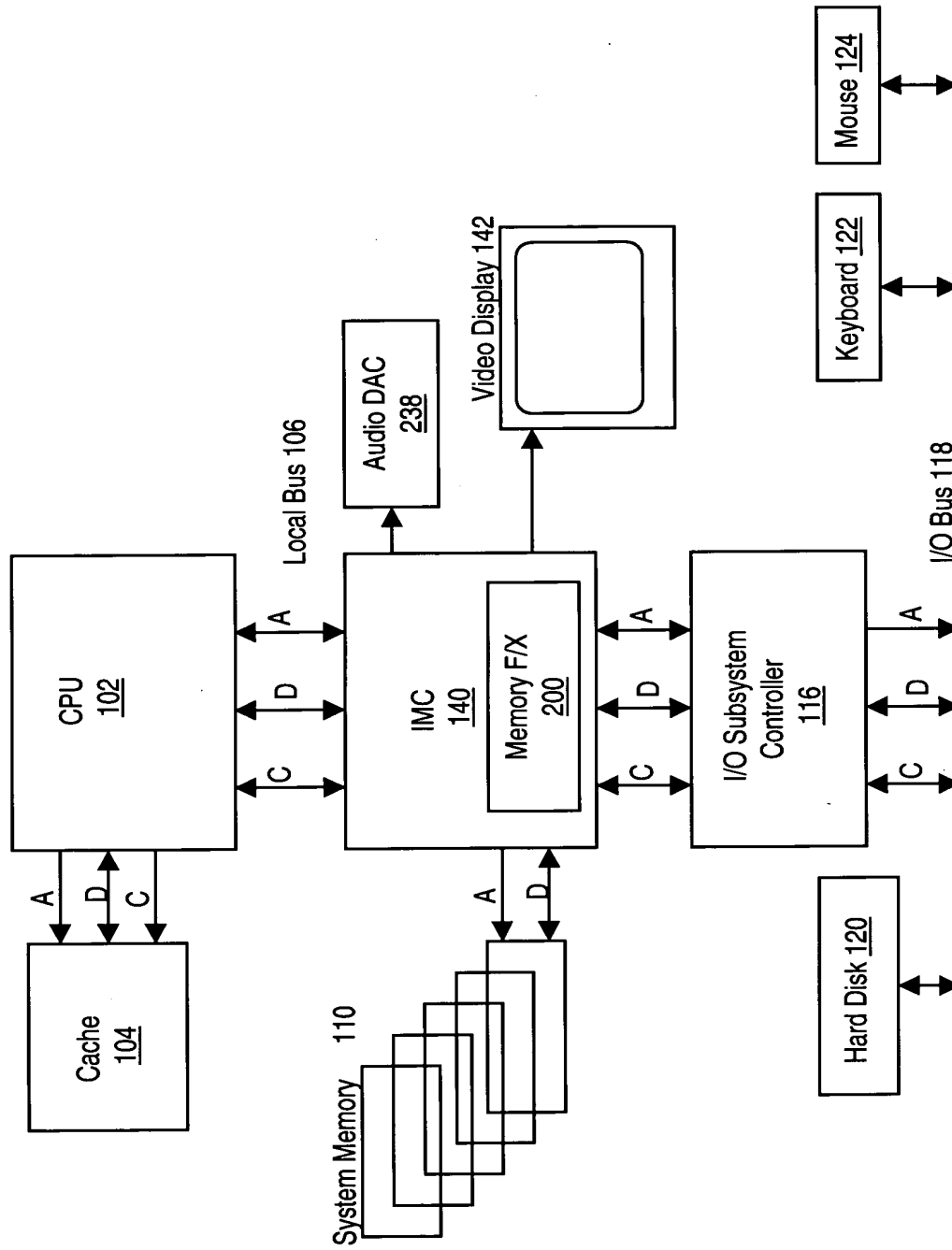


Figure 2a

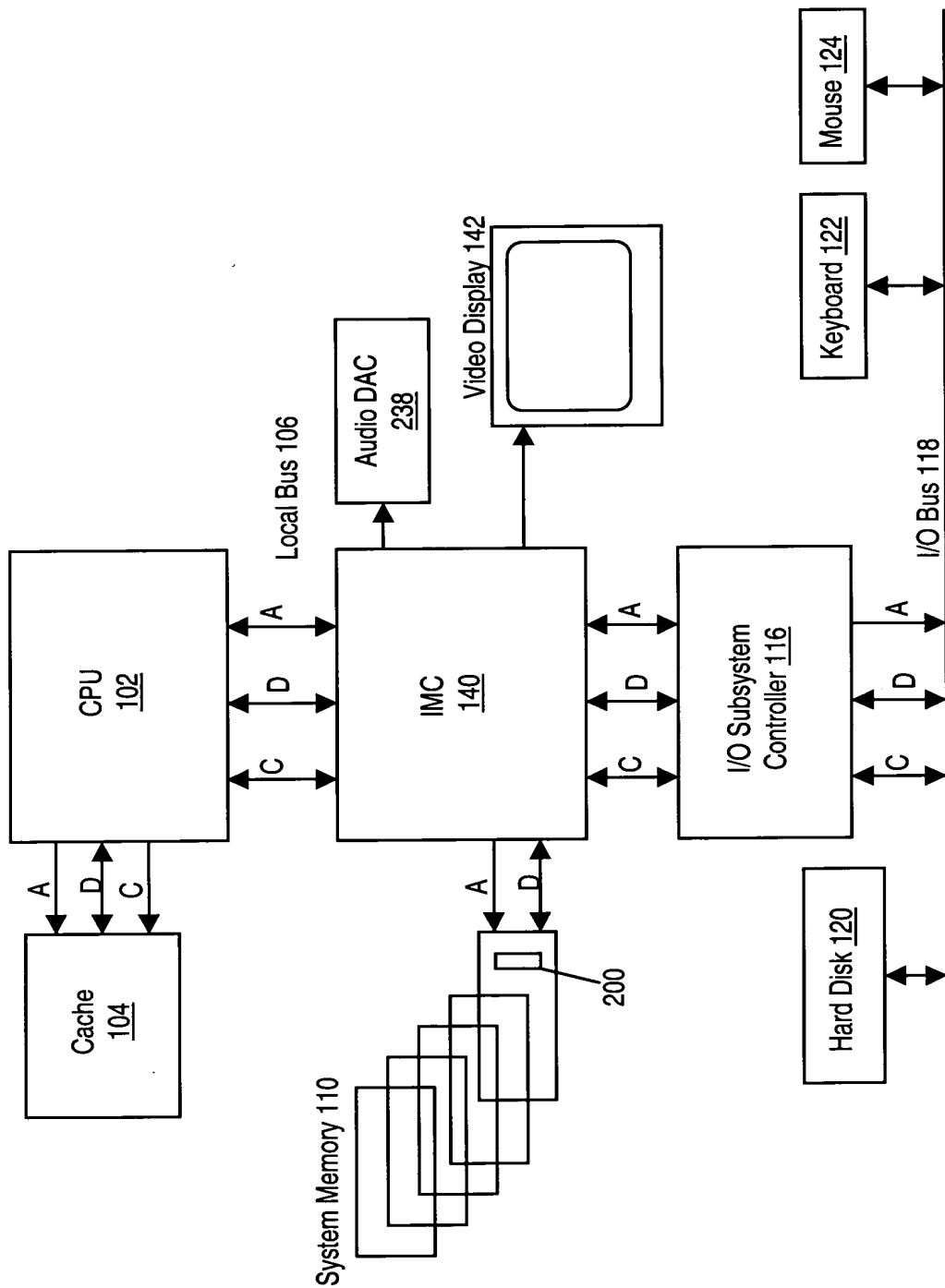


Figure 2b

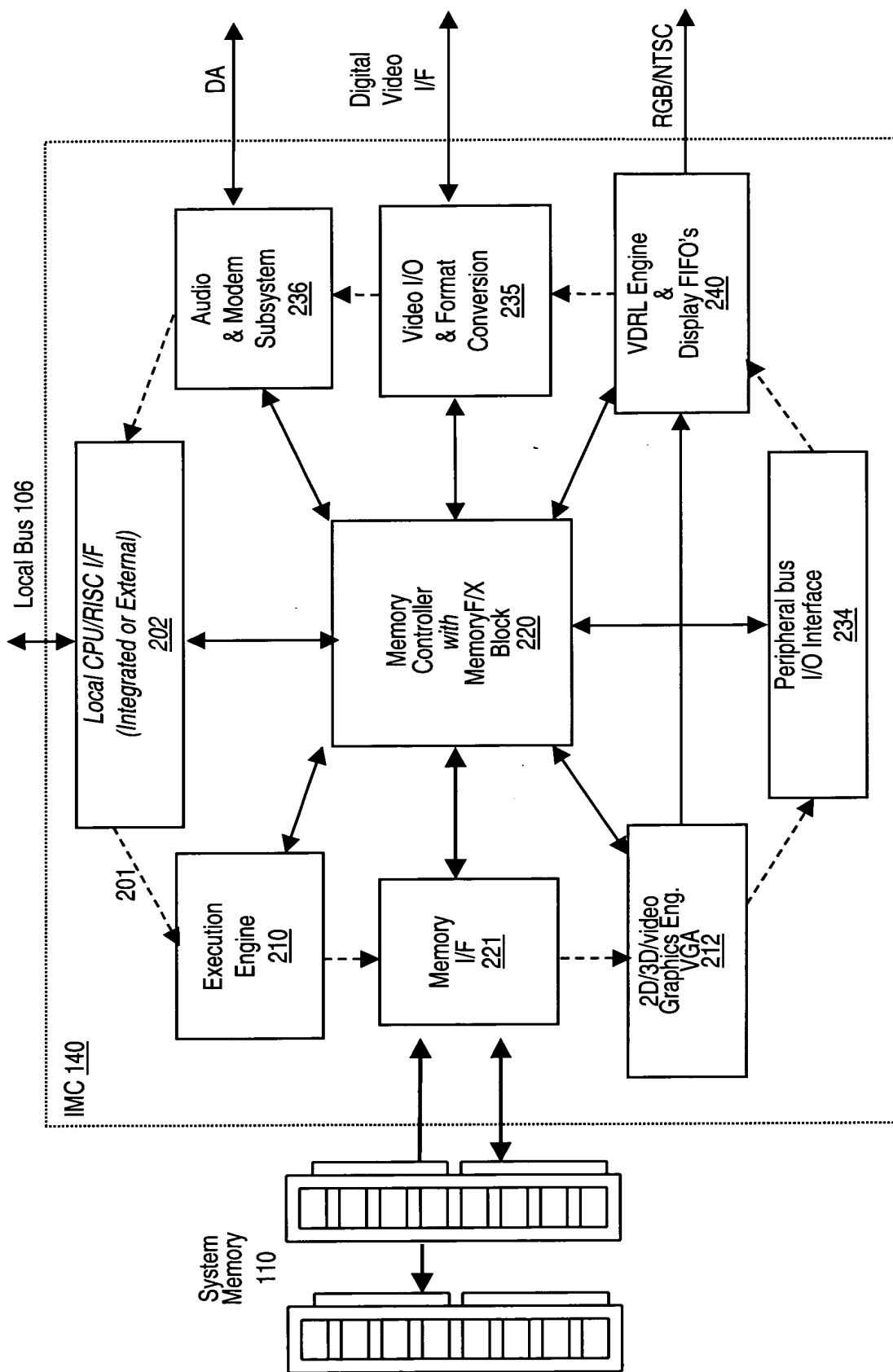


Figure 3

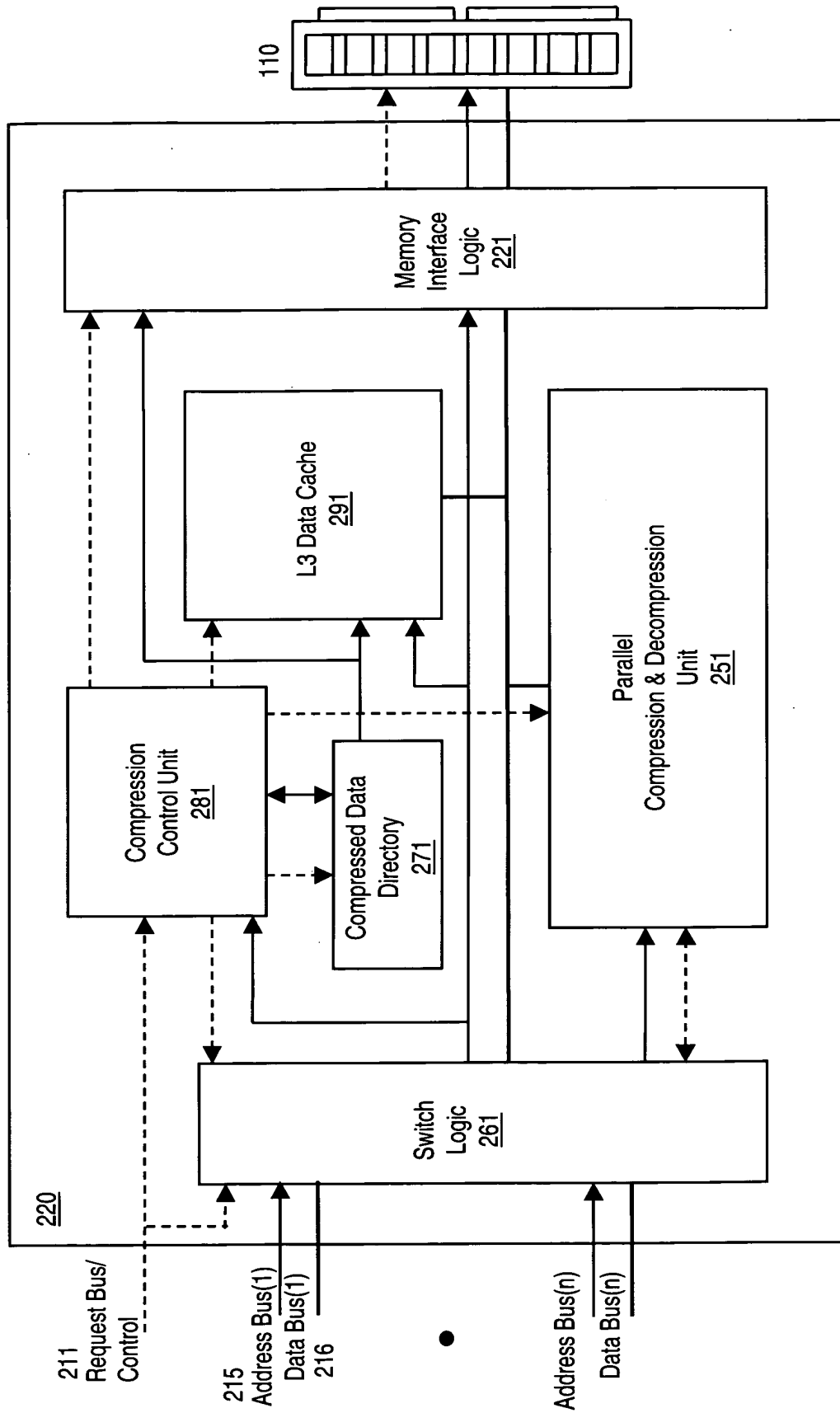


Figure 4

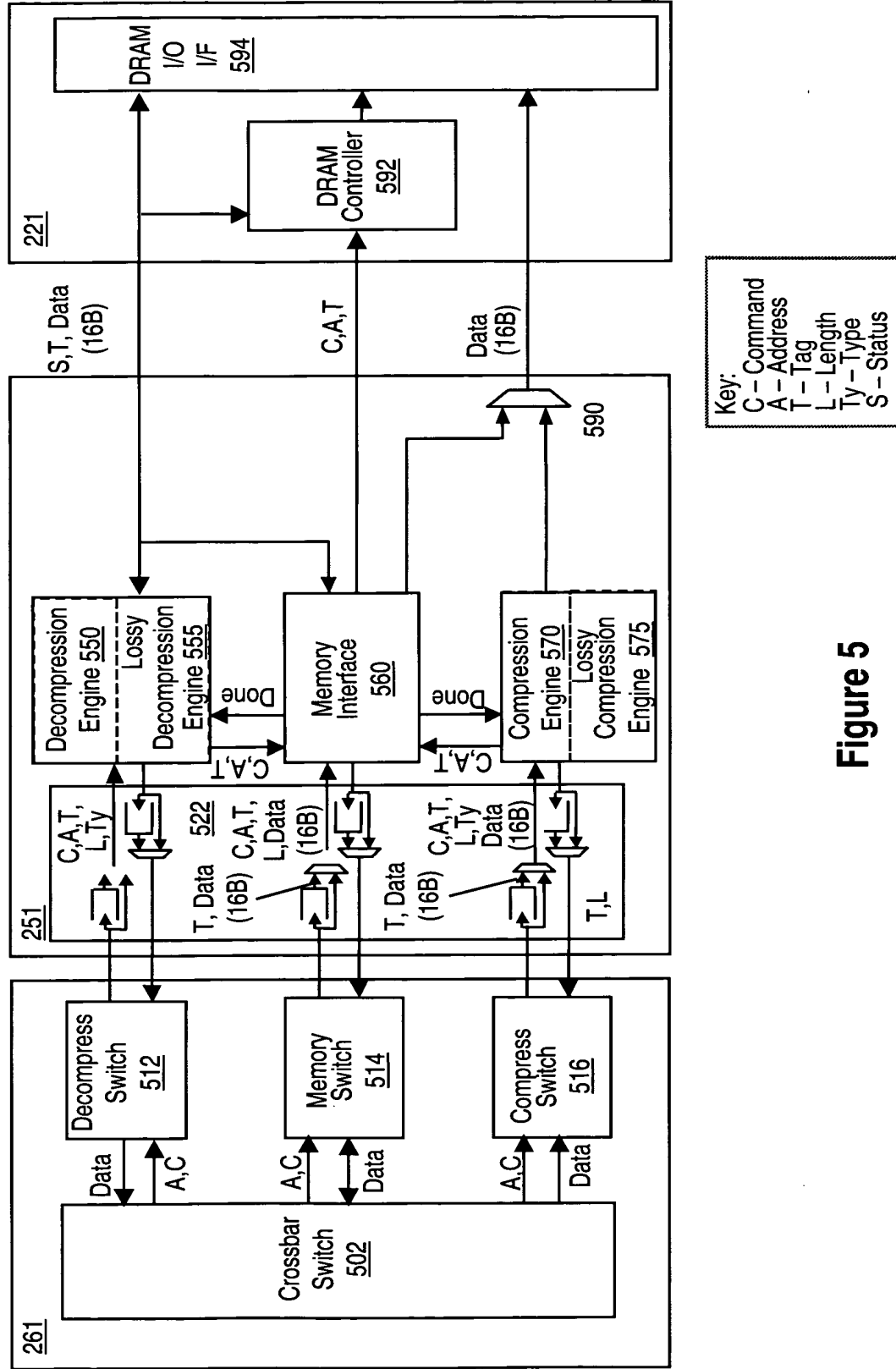


Figure 5

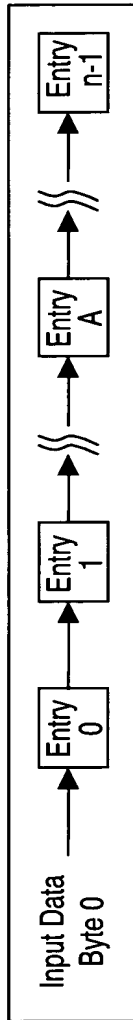


Figure 6a
(Prior Art)

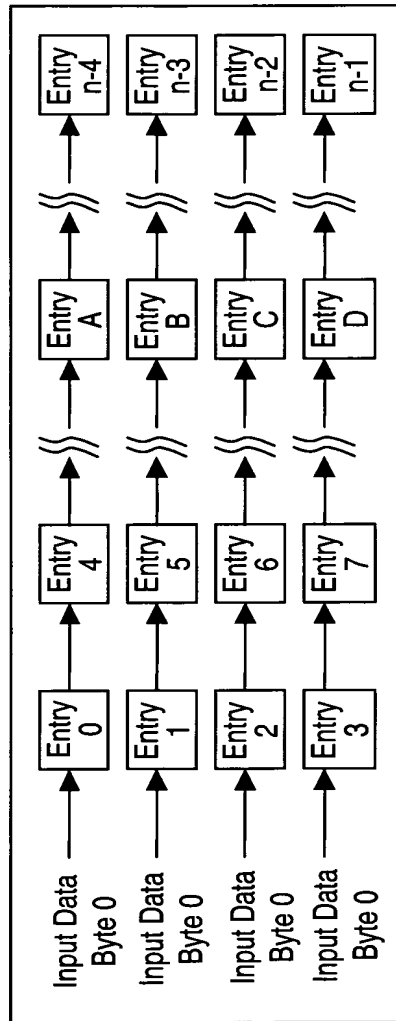


Figure 6b
(New Art)

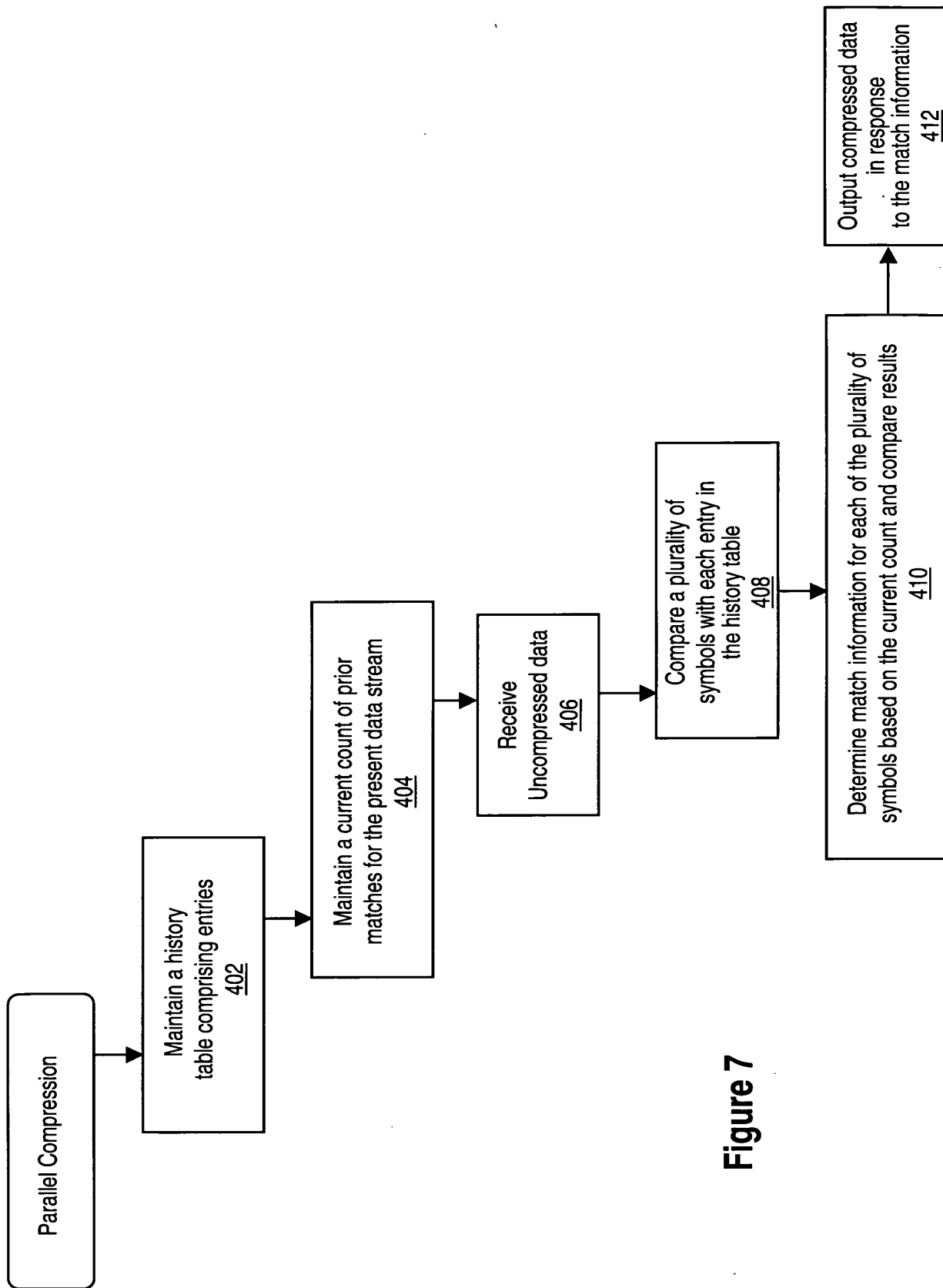
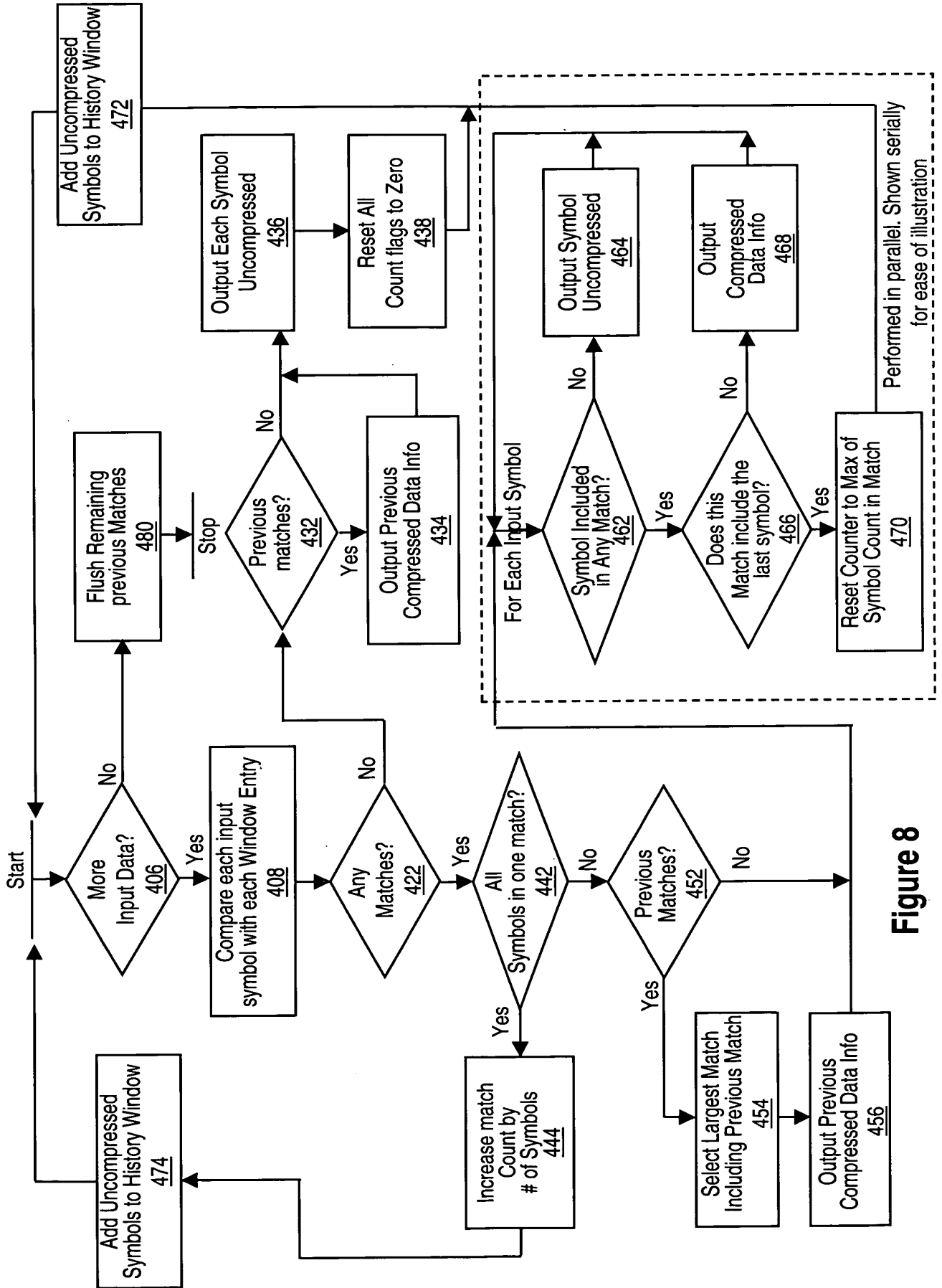


Figure 7



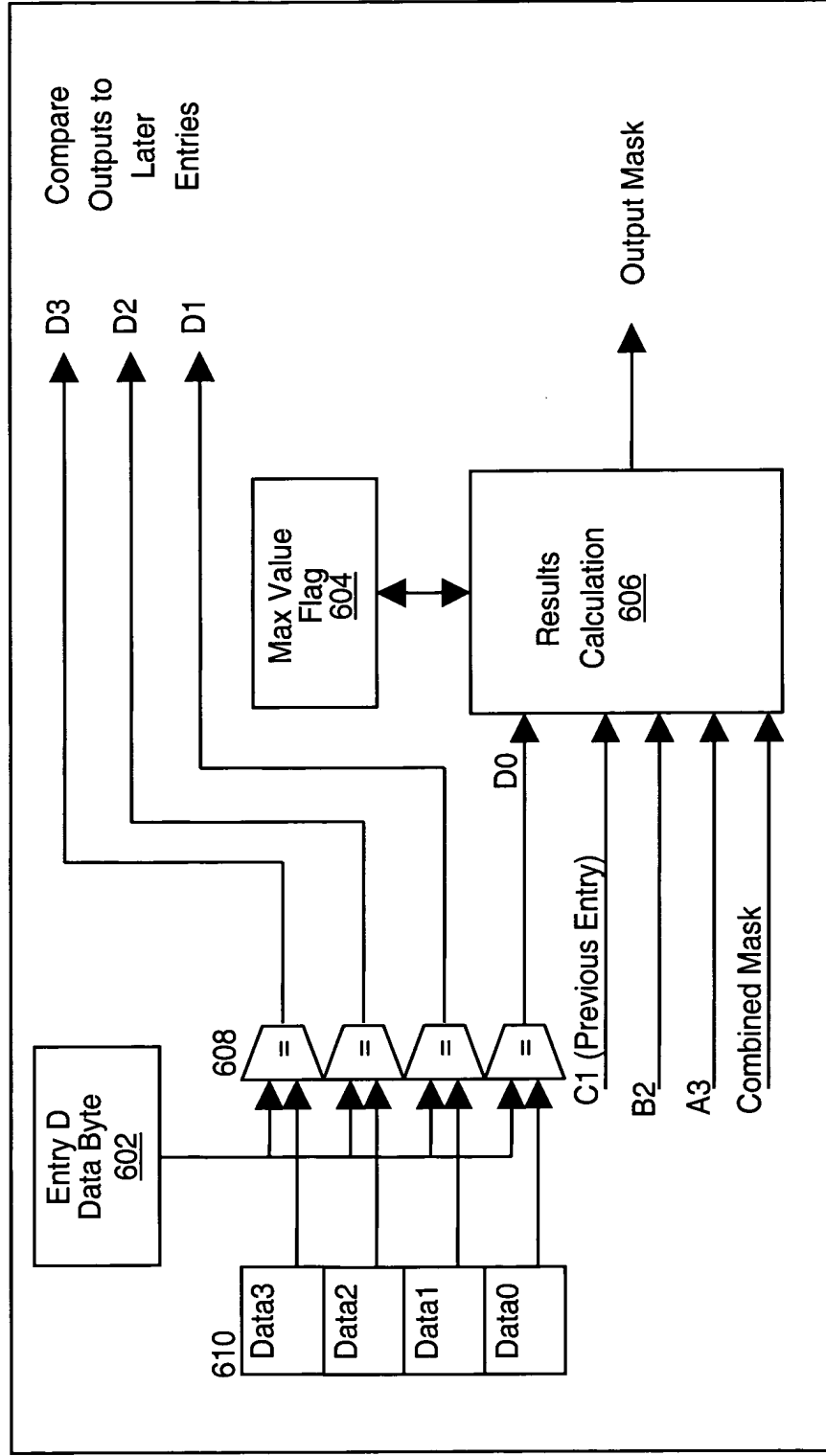


Figure 9

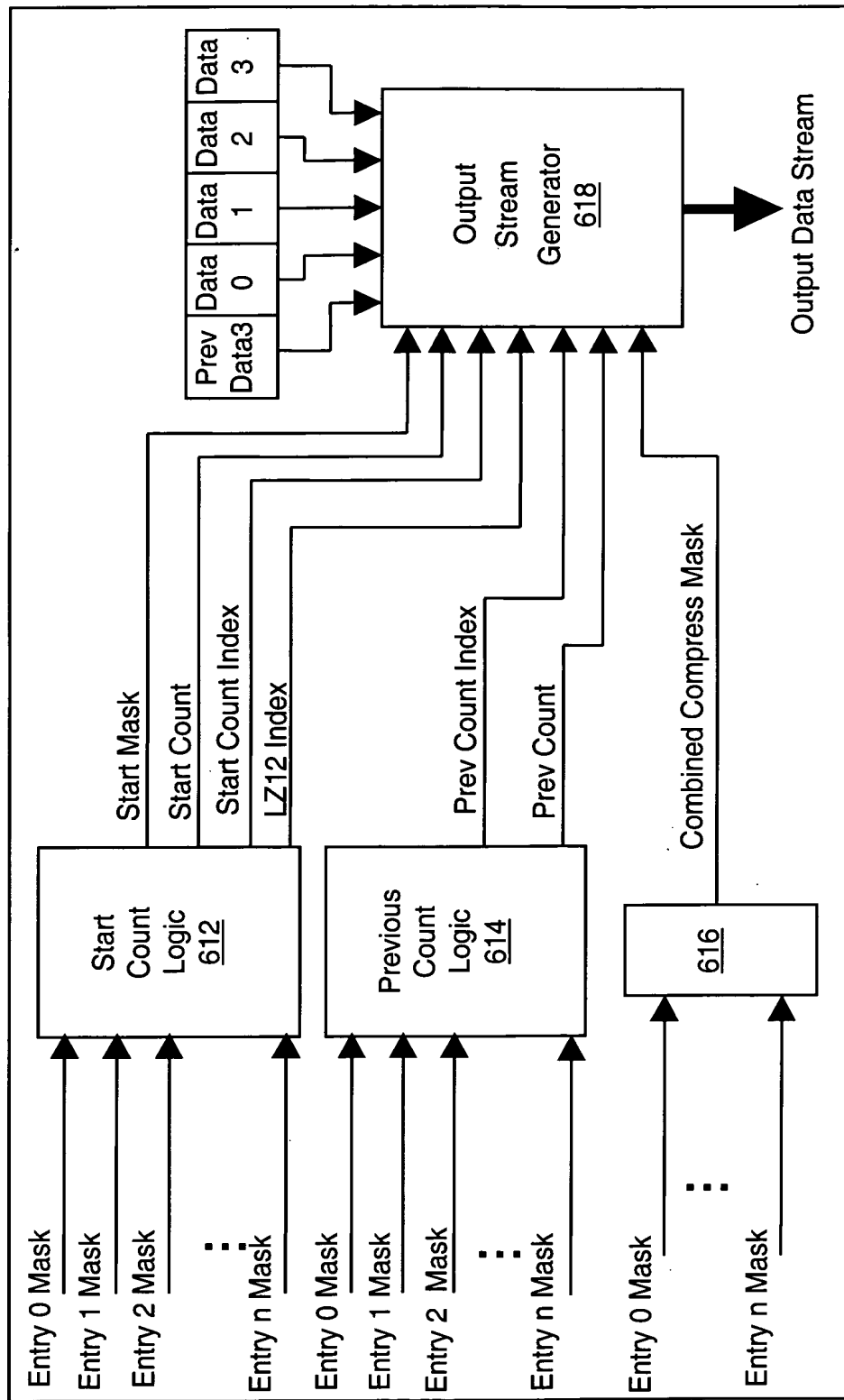


Figure 10

Input Matches				New Counter Value	Output Counter	Output Mask	Reset Value
D0	C1	B2	A3				
1	1	1	1	Saved+4	Saved +4	10000	0
1	1	1	0	0	Saved+3	10001	1
1	1	0	1	1	Saved+2	10010	2
1	1	0	0	0	Saved+2	10011	2
1	0	1	1	2	Saved+1	10100	3
1	0	1	0	0	Saved+1	10101	3
1	0	0	1	1	Saved+1	10110	3
1	0	0	0	0	Saved+1	10111	3
0	1	1	1	3	Saved	11000	4
0	1	1	0	0	Saved	01111	1
0	1	0	1	1	Saved	11010	4
0	1	0	0	0	Saved	11011	4
0	0	1	1	2	Saved	11100	4
0	0	1	0	0	Saved	11101	4
0	0	0	1	1	Saved	11110	4
0	0	0	0	0	Saved	11111	4

Fig. 11a

Input Matches				Output
D0	C1	B2	A3	Mask
1	1	1	1	1111
1	1	1	0	1110
1	1	0	1	1101
1	1	0	0	1100
1	0	1	1	1011
1	0	1	0	1010
1	0	0	1	1001
1	0	0	0	1000
0	1	1	1	0111
0	1	1	0	0110
0	1	0	1	0101
0	1	0	0	0100
0	0	1	1	0011
0	0	1	0	0010
0	0	0	1	0001
0	0	0	0	0000

Figure 11b

Combined Mask	Count
0000	0
0001	1
0010	0
0011	2
0100	0
0101	1
0110	0
0111	3
1000	0
1001	1
1010	0
1011	2
1100	0
1101	1
1110	0
1111	Count+4

Figure 11c

Output Masks	Combined Mask
M1234 4321 &~M1	0001
432 & ~M12	0010
43 & ~M123	0100
4 & ~M1234	1000
First valid row determines Combined Mask Output	
M-Max Count Flag 1-1 st Symbol Match 2-2 nd Symbol Match 3-3 rd Symbol Match 4-4 th Symbol Match	

Figure 11d

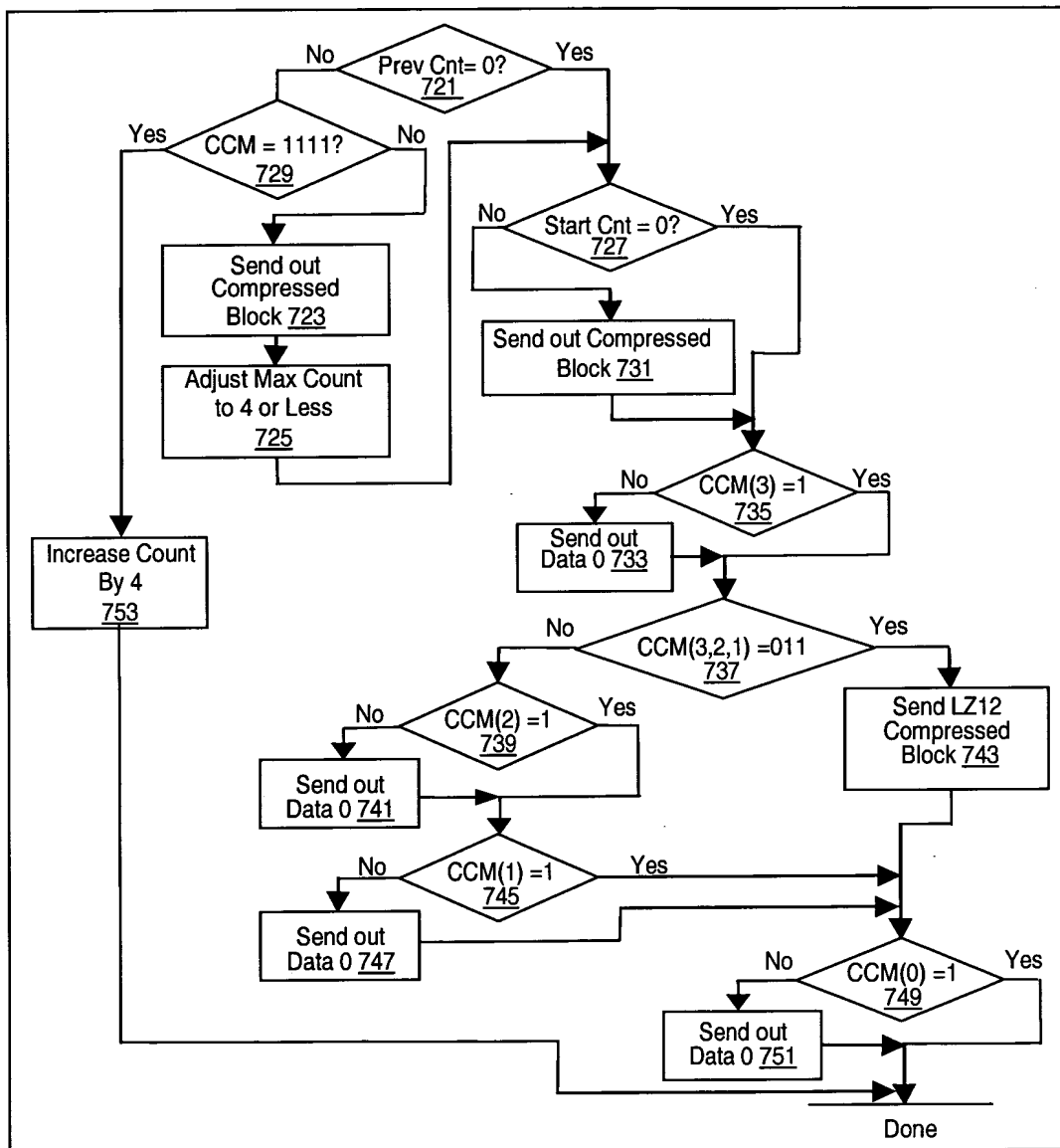


Figure 12

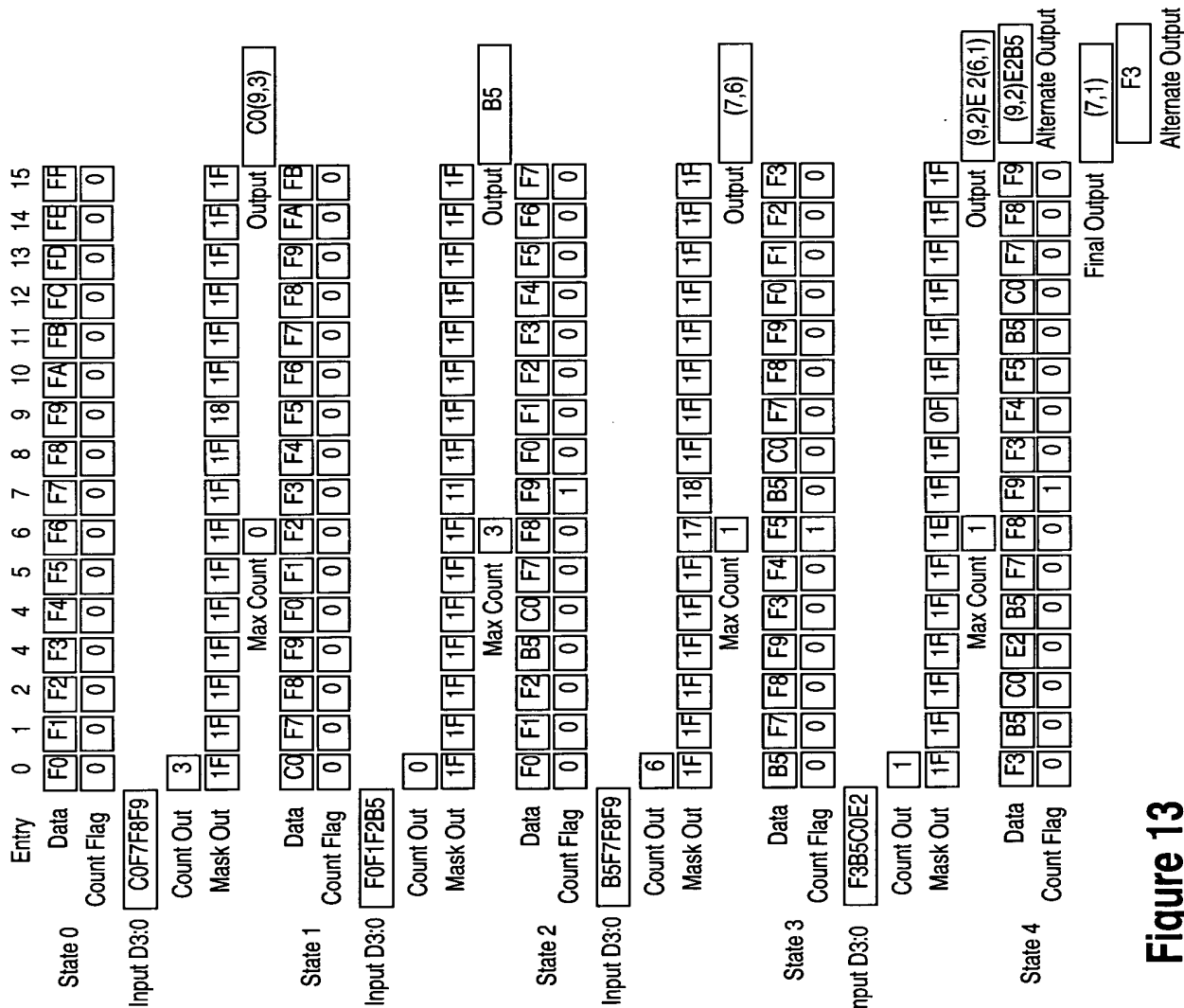
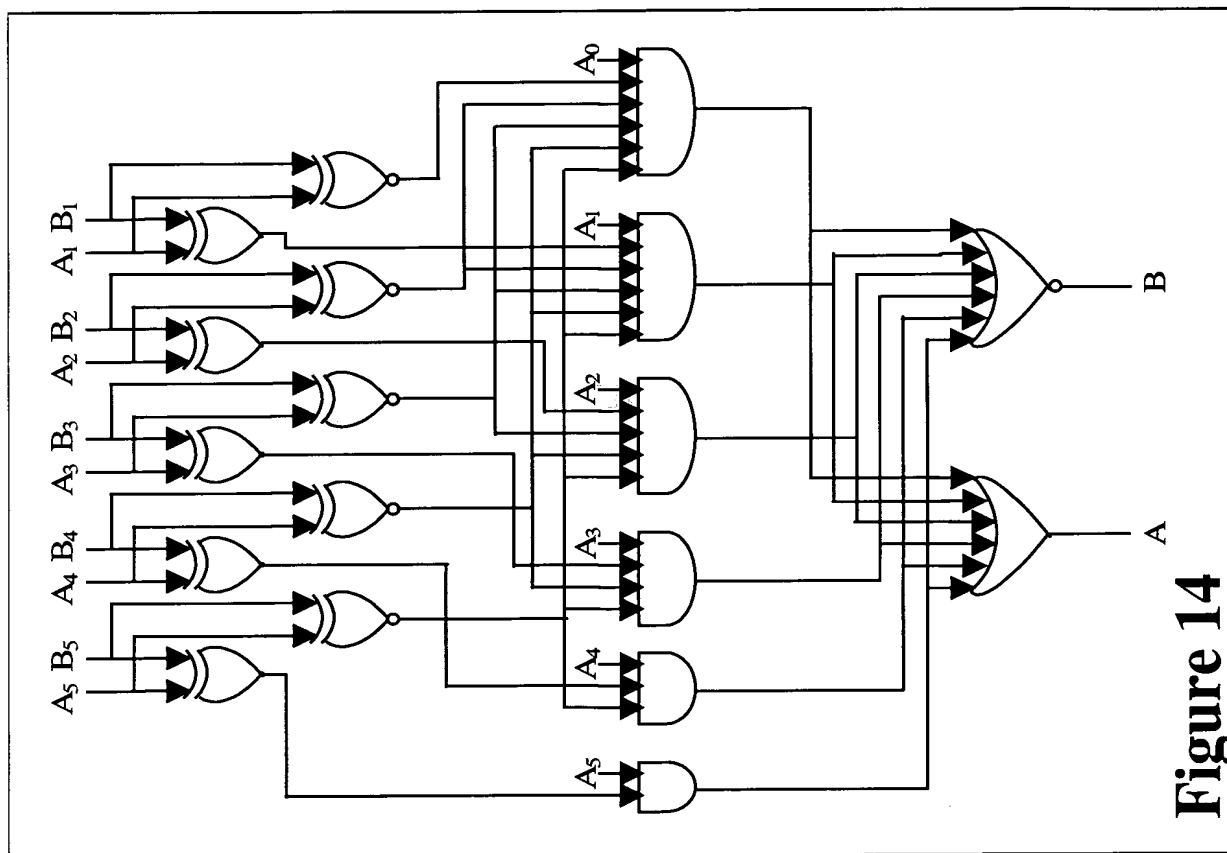


Figure 13



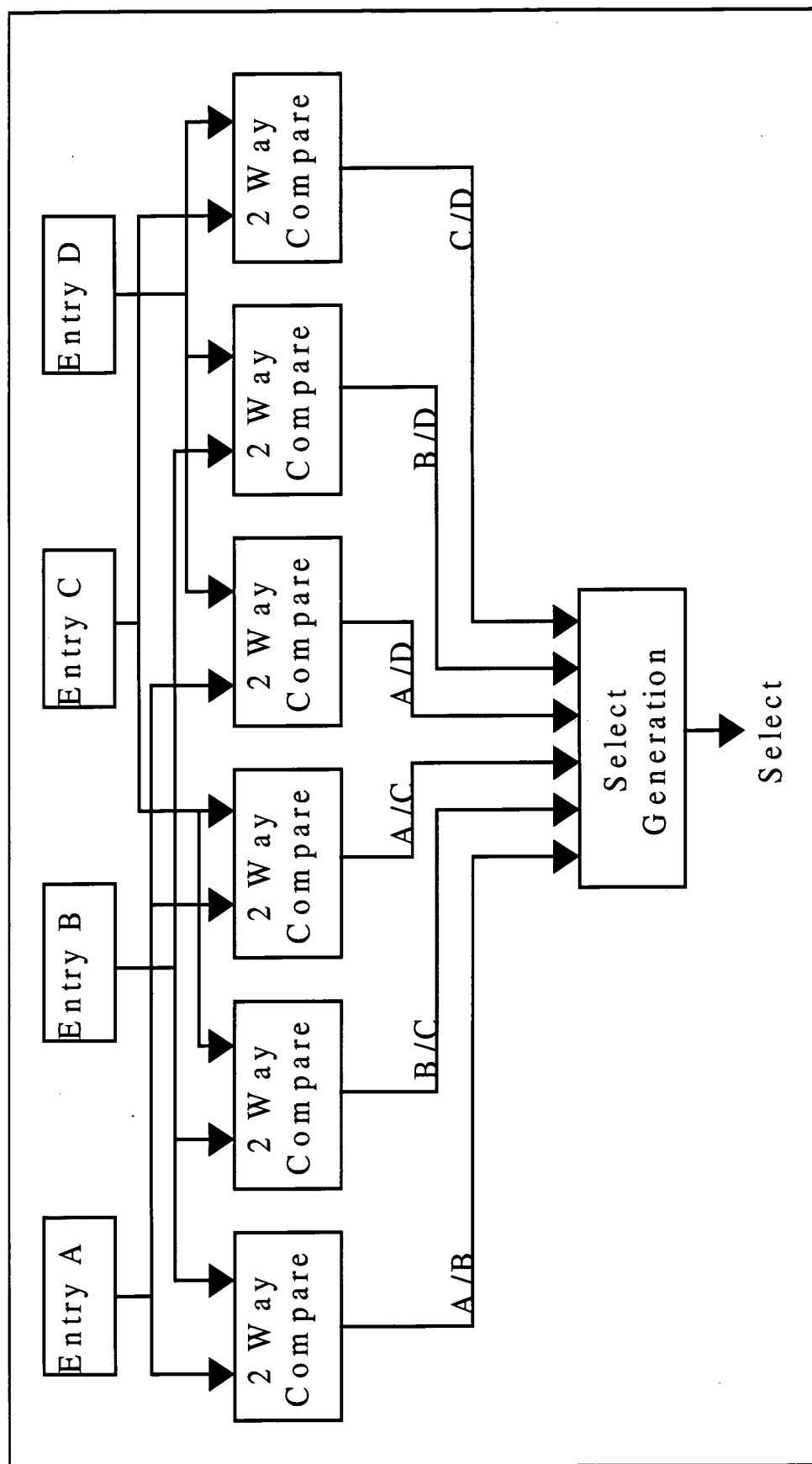
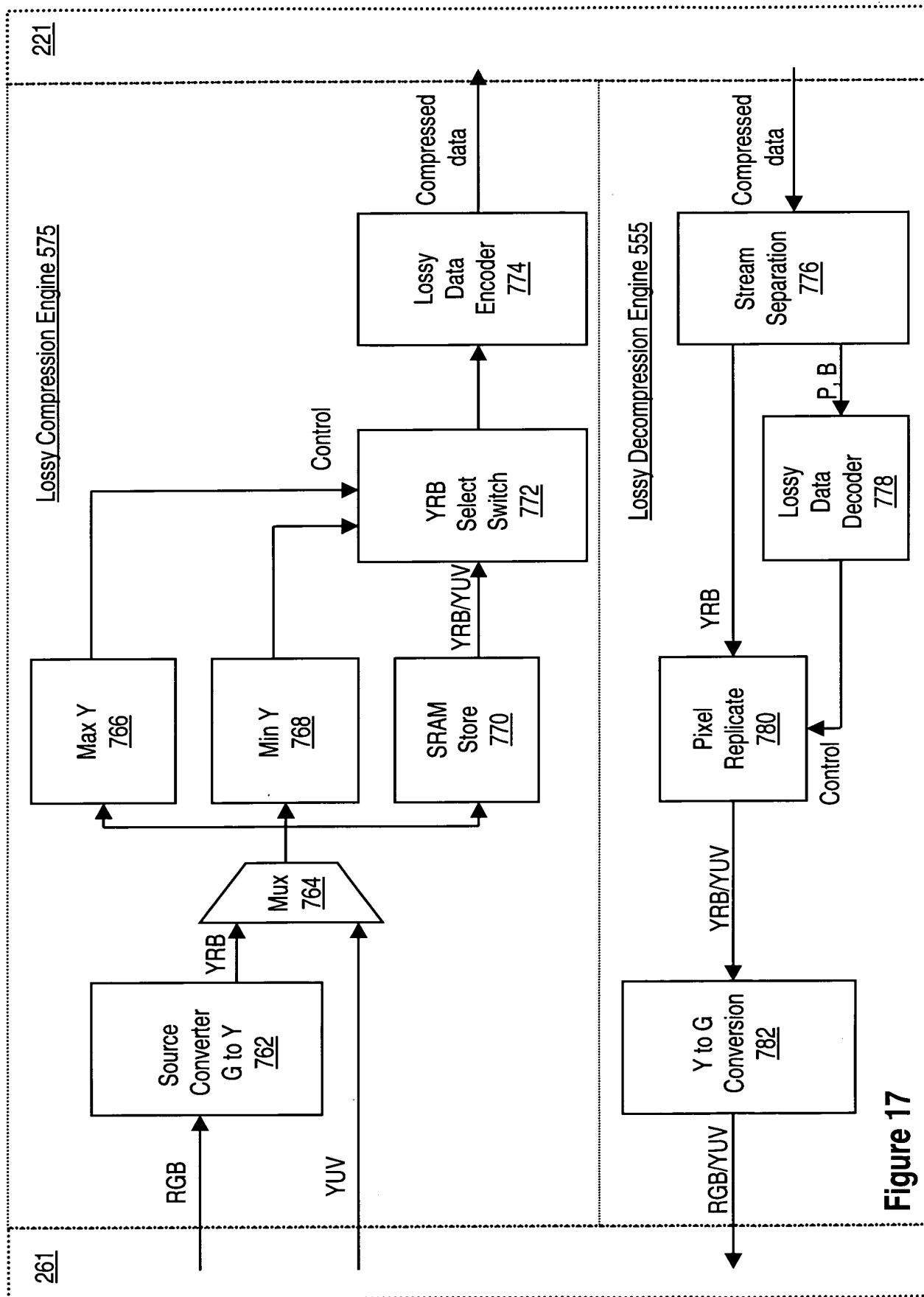


Figure 15

A/ B	B/ C	C/ D	D/ A	A/ C	B/ D	Output
0	X	X	1	0	X	A
1	0	X	X	X	0	B
X	1	0	X	1	X	C
X	X	1	0	X	1	D

Figure 16



Ymax = Ymin	1 color	Ymax	Ymax	Rmax	Bmax	11			3 Bytes
		6 bits	6 bits	5 bits	5 bits	2 bits			
Ymax != Ymin	2 colors	Ymax	Ymin	Rmax	Rmin	Bmax	Bmin	P bits	6 Bytes
		6 bits	6 bits	5 bits	5 bits	5 bits	5 bits	16 bits	
Ymax != Ymin	>2 colors	Ymin	Ymax	Rmax	Rmin	Bmax	Bmin	P bits	8 Bytes
		6 bits	6 bits	5 bits	5 bits	5 bits	5 bits	32 bits	

Figure 18

Ymax = Ymin	Amax = Amin = 0x00	1 color	Ymax	Ymax	Rmax	Bmax	00				3 Bytes
			6 bits	6 bits	5 bits	5 bits	2 bits				
Ymax = Ymin	Amax = Amin = 0xFF	1 color	Ymax	Ymax	Rmax	Bmax	11				3 Bytes
			6 bits	6 bits	5 bits	5 bits	2 bits				
Ymax = Ymin	Amax = Amin != 00 or FF	1 color	Ymax	Ymax	Rmax	Bmax	01	Amax	Amin		4/5 Bytes
			6 bits	6 bits	5 bits	5 bits	2 bits	4/8 bits	4/8 bits		
Ymax = Ymin	Amax != Amin	1 color	Ymax	Ymax	Rmax	Bmax	01	Amax	Amin	P bits	6/7 Bytes
		2 Alphas	6 bits	6 bits	5 bits	5 bits	2 bits	4/8 bits	4/8 bits	16 bits	
Ymax = Ymin	Amax != Amin	1 color	Ymax	Ymax	Rmax	Bmax	10	Amax	Amin	P bits	8/9 Bytes
		>2 Alphas	6 bits	6 bits	5 bits	5 bits	2 bits	4/8 bits	4/8 bits	32 bits	
Ymax != Ymin	X	2 colors	Ymax	Ymin	Rmax	Rmin	Bmax	Bmin	Amax	Amin	P bits
			6 bits	6 bits	5 bits	5 bits	5 bits	4/8 bits	4/8 bits	16 bits	
Ymax != Ymin	X	>2 colors	Ymin	Ymax	Rmax	Rmin	Bmax	Bmin	Amax	Amin	P bits
			6 bits	6 bits	5 bits	5 bits	5 bits	4/8 bits	4/8 bits	32 bits	

Figure 19

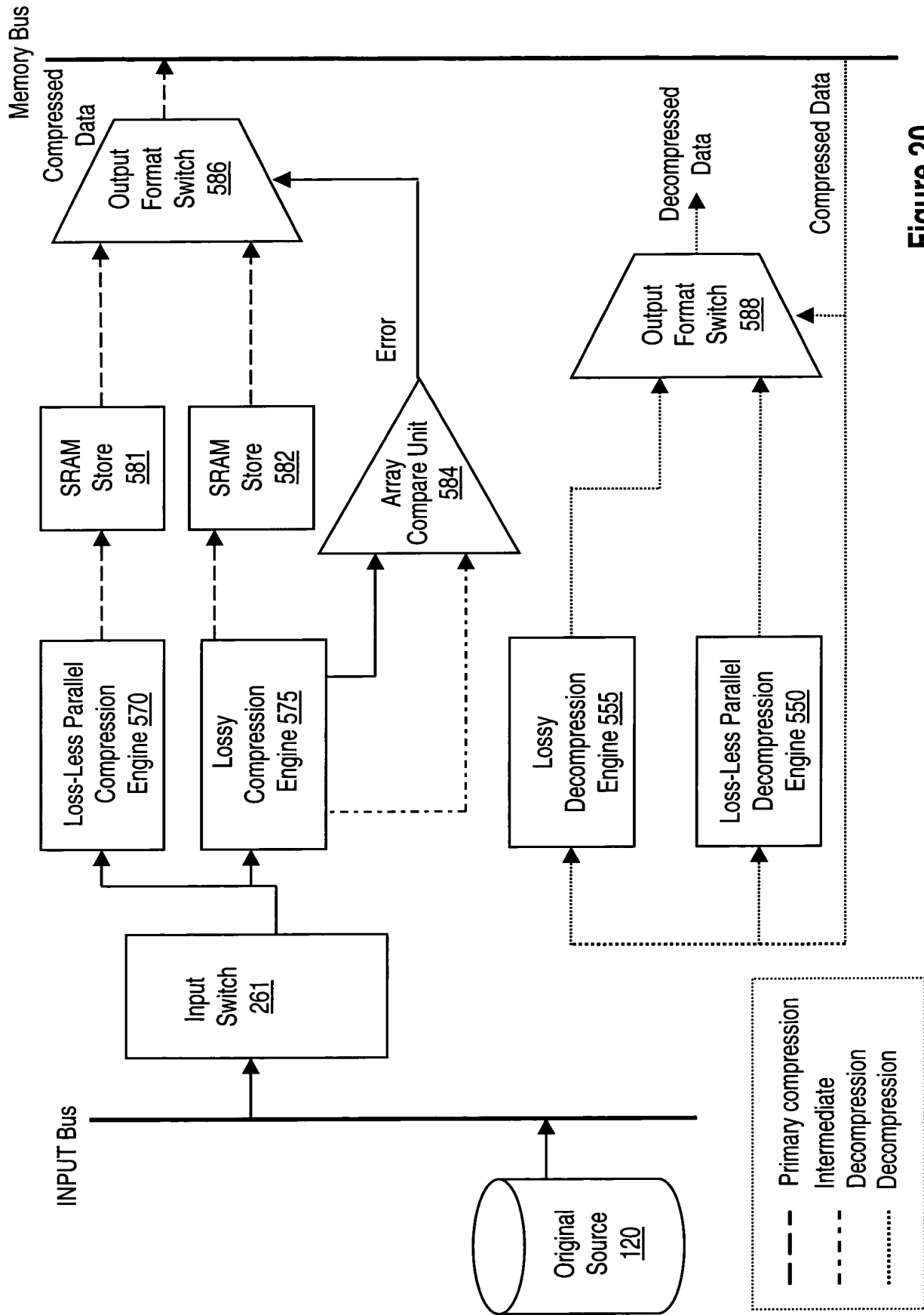


Figure 20

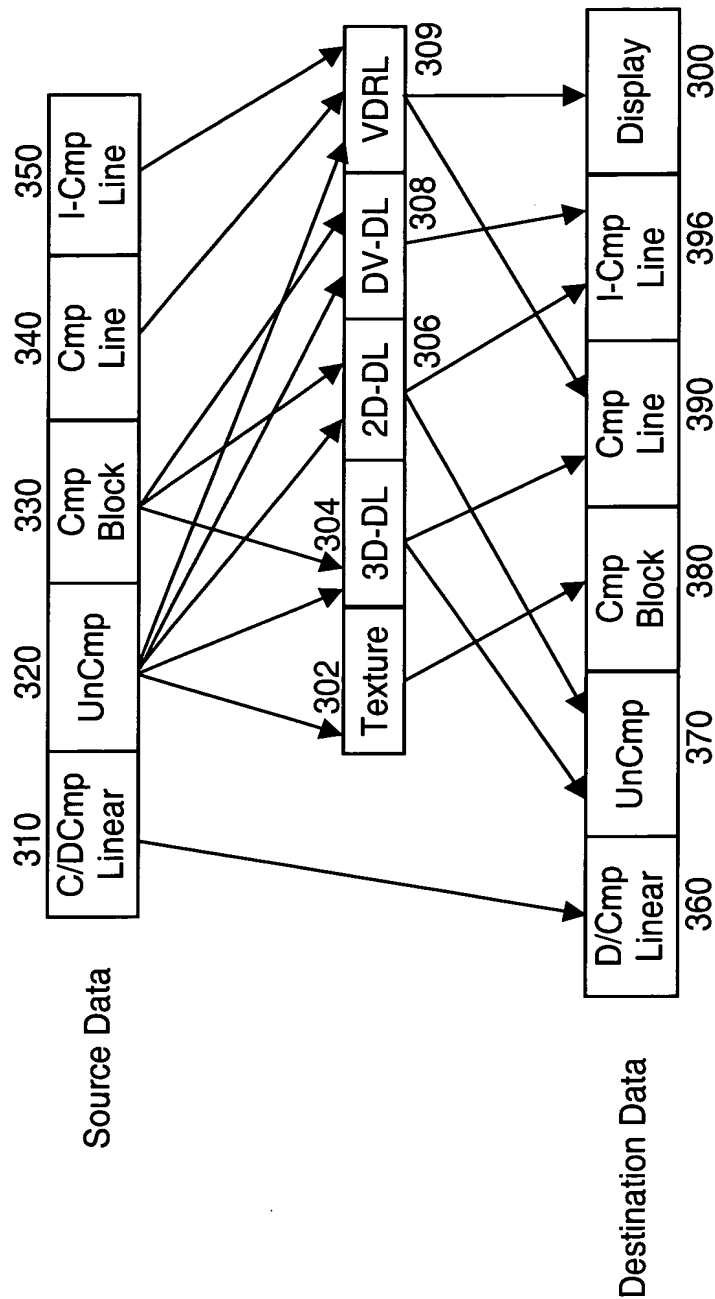


Figure 21

Compression of
received data



Receive
uncompressed
data
802



Determine a compression
mode of the data
804



Selectively compress
the uncompressed
data according to the
compression mode
806



Store the data in the
memory; store compression mode
information with the data
808

Figure 22

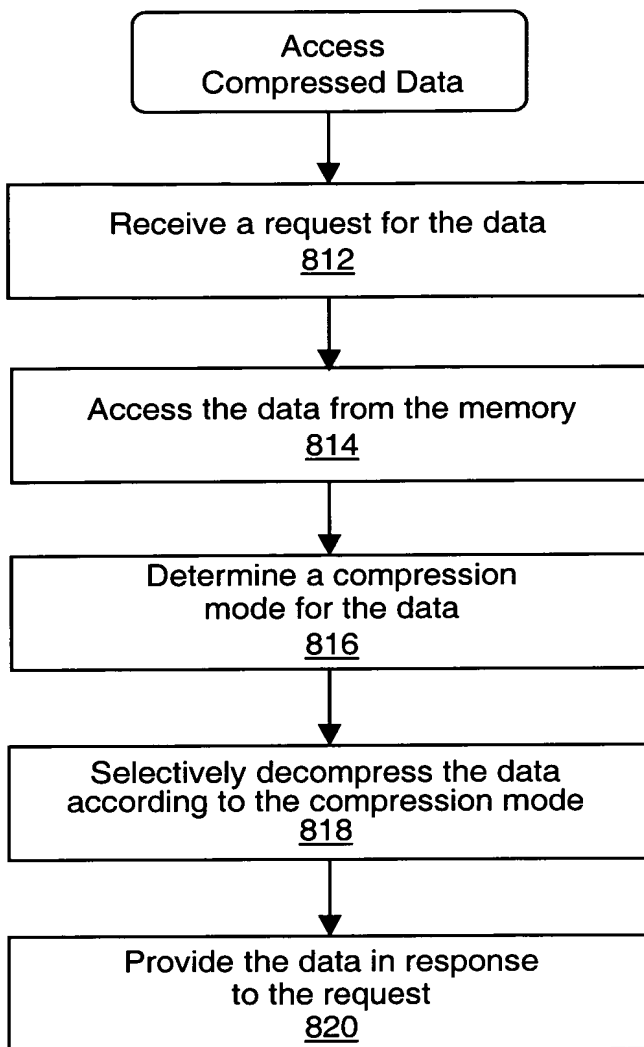


Figure 23

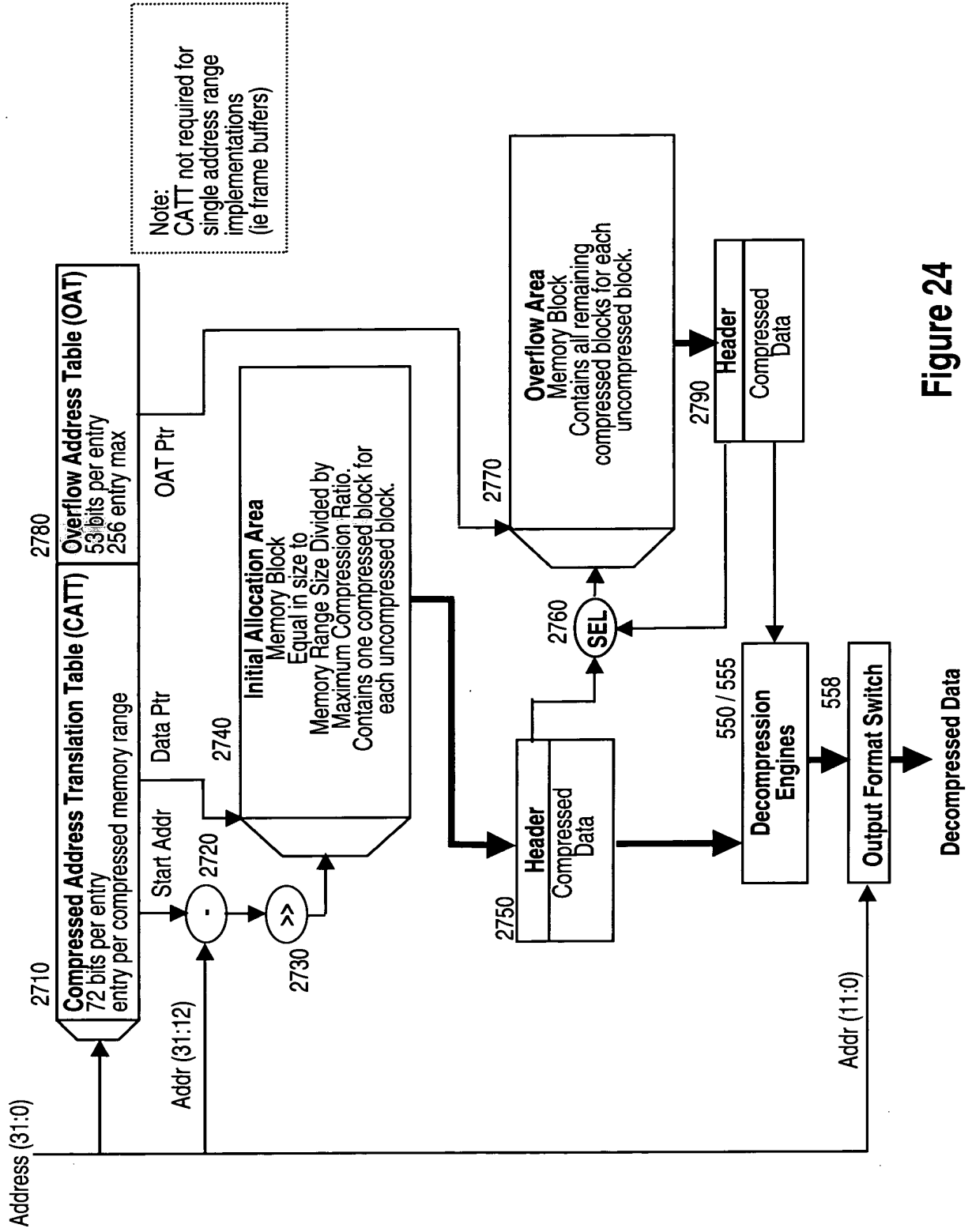


Figure 24

Memory Allocation Fields

Compressed Address Translation Table (CATT) – 128 Entry Design Limit				
Starting Addr	Ending Addr	Type	Data Ptr	OAT Ptr
20 bits	20 bits	4 bits	20 bits	8 bits
4GB Addressability		Compressed		
4K Boundry	4K Boundry	Blk Size	4K Boundry	4K Boundry
Overflow Address Table (OAT) – 256 Entry Max				
Overflow Ptr	Next Block Ptr	Next OAT Ptr		Next OAT Valid
20 bits	24 bits	8 bits		1 bit
4 GB Addressability		Points to next entry		
4K Boundry		in this table		
Initial Header Description				
Value	# of bits	Meaning		
0	1	Last Block/Unused		
10 A (20 bits)	22	The next block is at offset A in the Overflow Area		
11 1A(8+20 bits)	30	The next block is at offset A in the Overflow Area of OAT entry I		
Overflow Header Description				
Value	# of bits	Meaning		
00	2	Last Block/Unused		
01	2	The next block follows physically after this one		
10A (8 bits)	10	The next block is A blocks before this one (or after?)		
110A (20 bits)	23	The next block is at offset A in the Overflow Area		
111 1A (8+20 bits)	31	The next block is at offset A in the Overflow Area of OAT entry I		

Figure 25

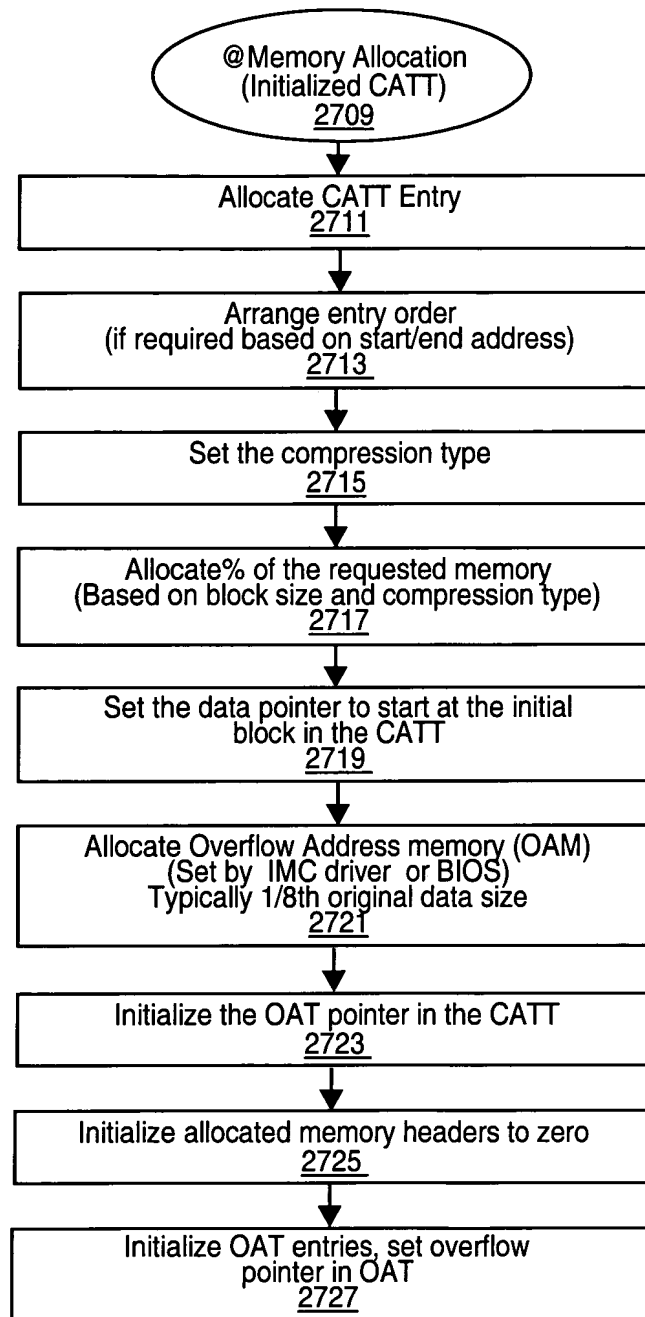


Figure 26

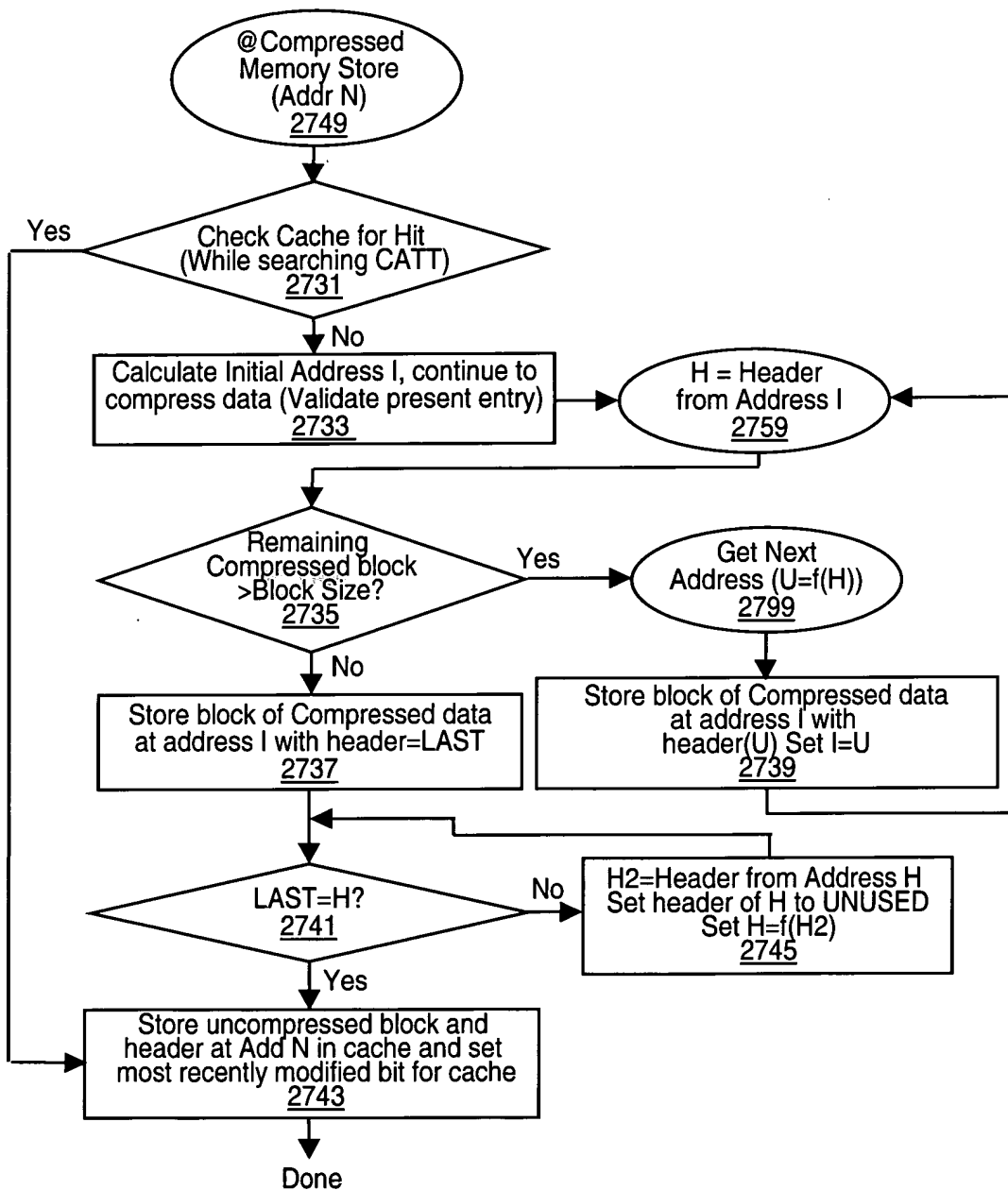


Figure 27

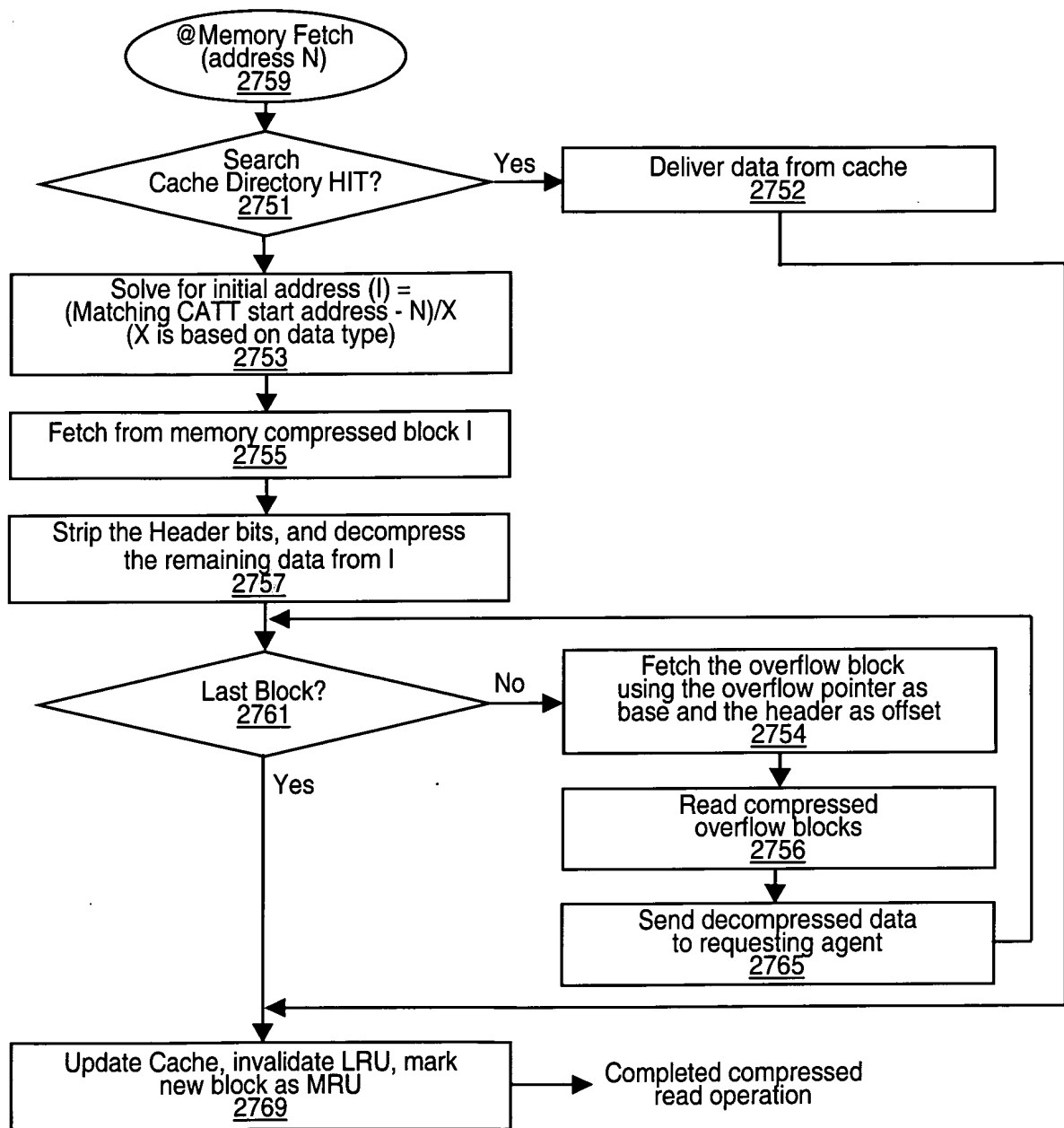


Figure 28

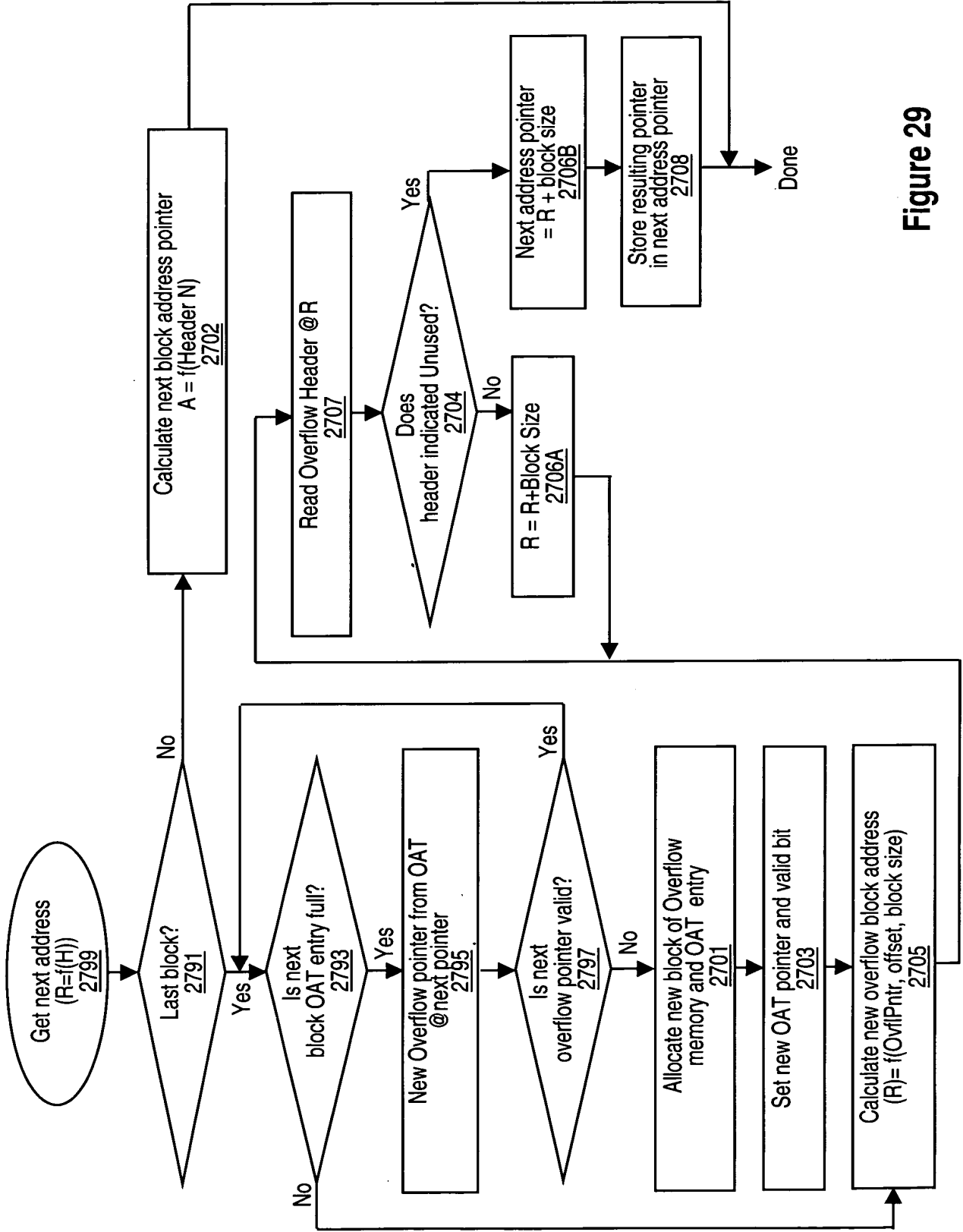


Figure 29

Uncomp Block Bytes	Type	Initial Block Size Bytes	Overflow Block Size Bytes	Max Comp Ratio (X:1)	Initial Allocation	Header w/o OF	Header w/ OF Non-Frag	Header w/ OF Fragmented
4096	8	256	64	16	6%	0.0%	0.4%	4.1%
2048	7	128	64	16	6%	0.1%	0.5%	4.2%
1024	6	64	64	16	6%	0.2%	0.6%	4.3%
512	5	64	64	8	13%	0.2%	0.9%	4.3%
256	4	64	64	4	25%	0.2%	1.4%	4.3%
128	3	32	32	4	25%	0.4%	2.8%	8.8%
64	2	32	16	2	50%	0.4%	5.1%	13.6%
32	1	32	8	1	100%	0.4%	8.9%	11.5%

Figure 30

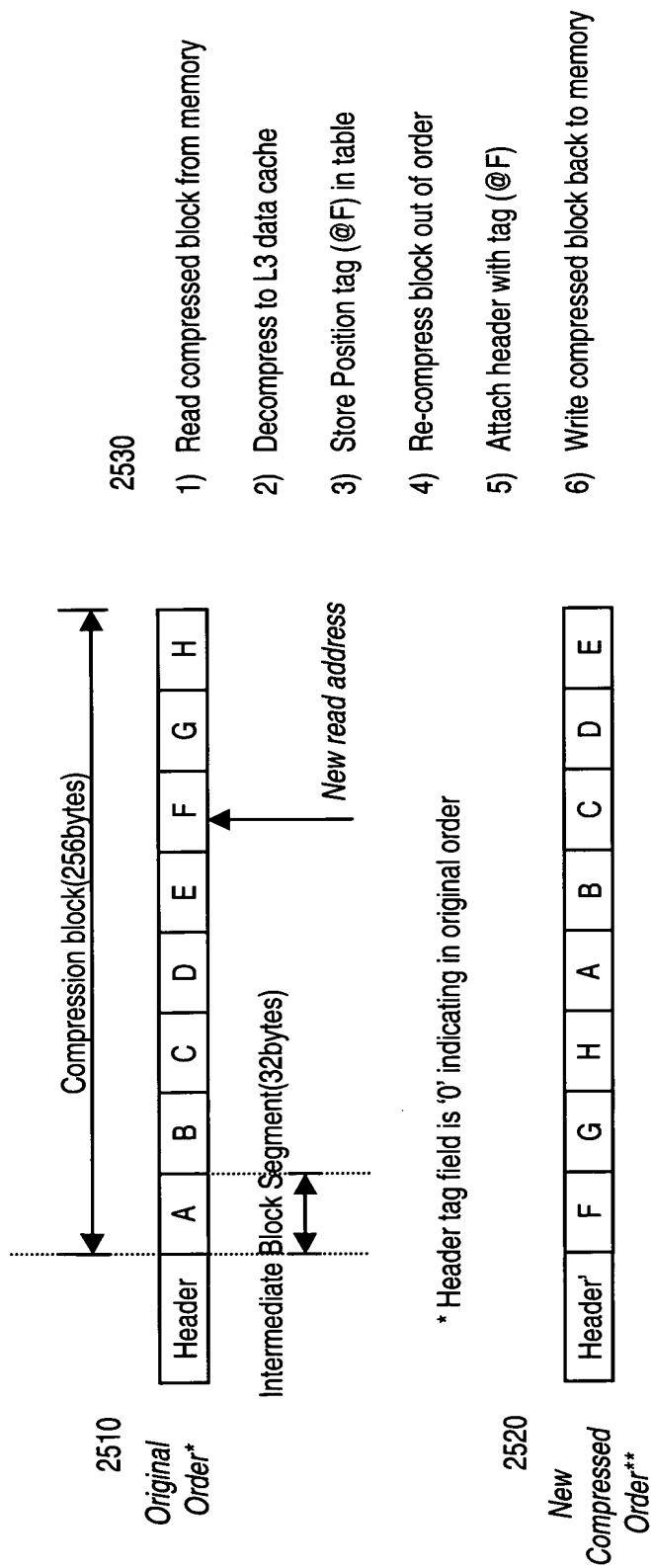


Figure 31

Bytes Compressed	Flag	Index	Count	Data	Bits Used
0	0	-	-	8b	9
1	10	6b	-	-	8
2	1100	6b	-	-	10
3	1101	6b	-	-	10
4	1110	6b	-	-	10
5	1111000	6b	-	-	13
6	1111001	6b	-	-	13
7	1111010	6b	-	-	13
8	1111011	6b	-	-	13
9	1111100	6b	-	-	13
10	1111101	6b	-	-	13
11	1111110	6b	-	-	13
>11	1111111	6b	12b	-	25

Figure 32

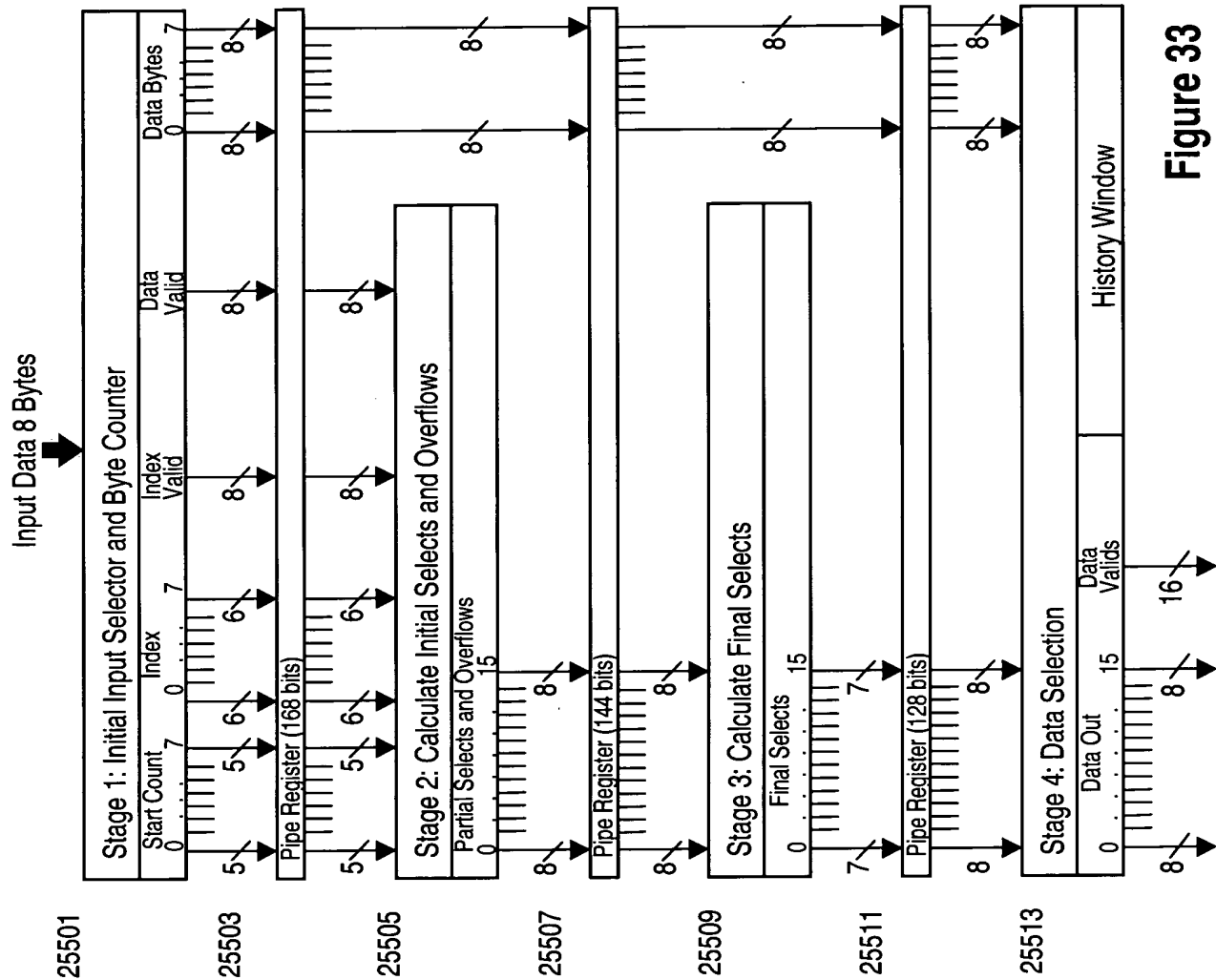
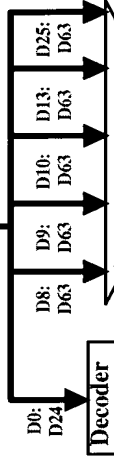
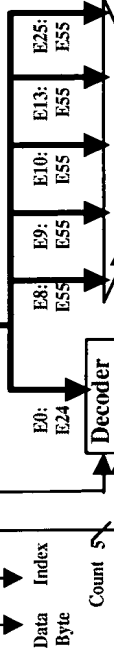


Figure 33

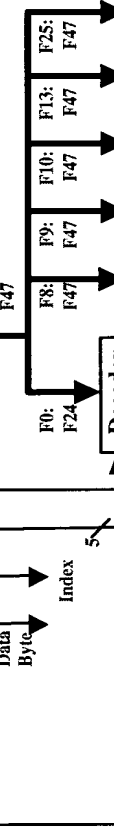
25521



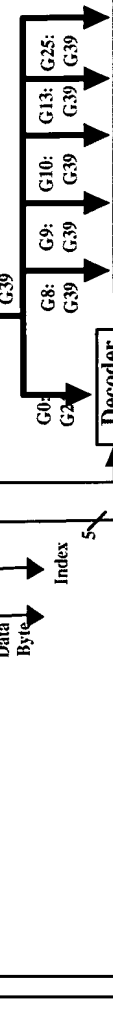
25523



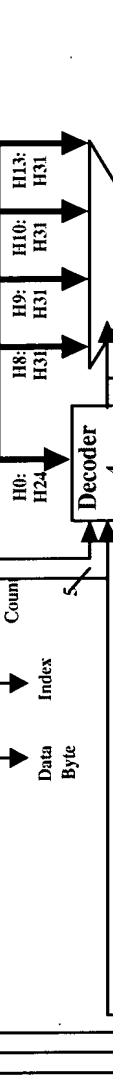
25525



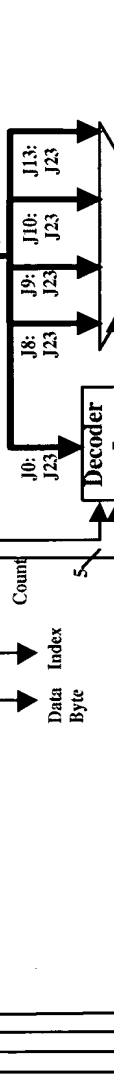
25527



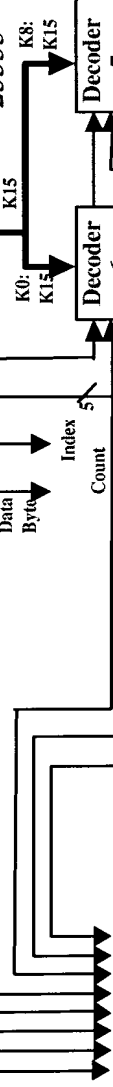
25529



25531



25533



25535

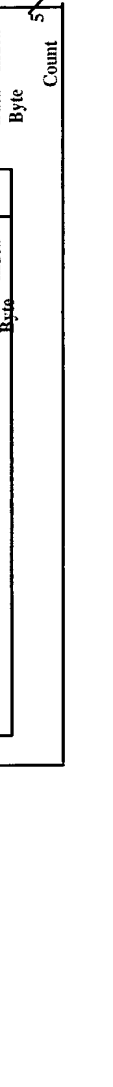


Figure 34

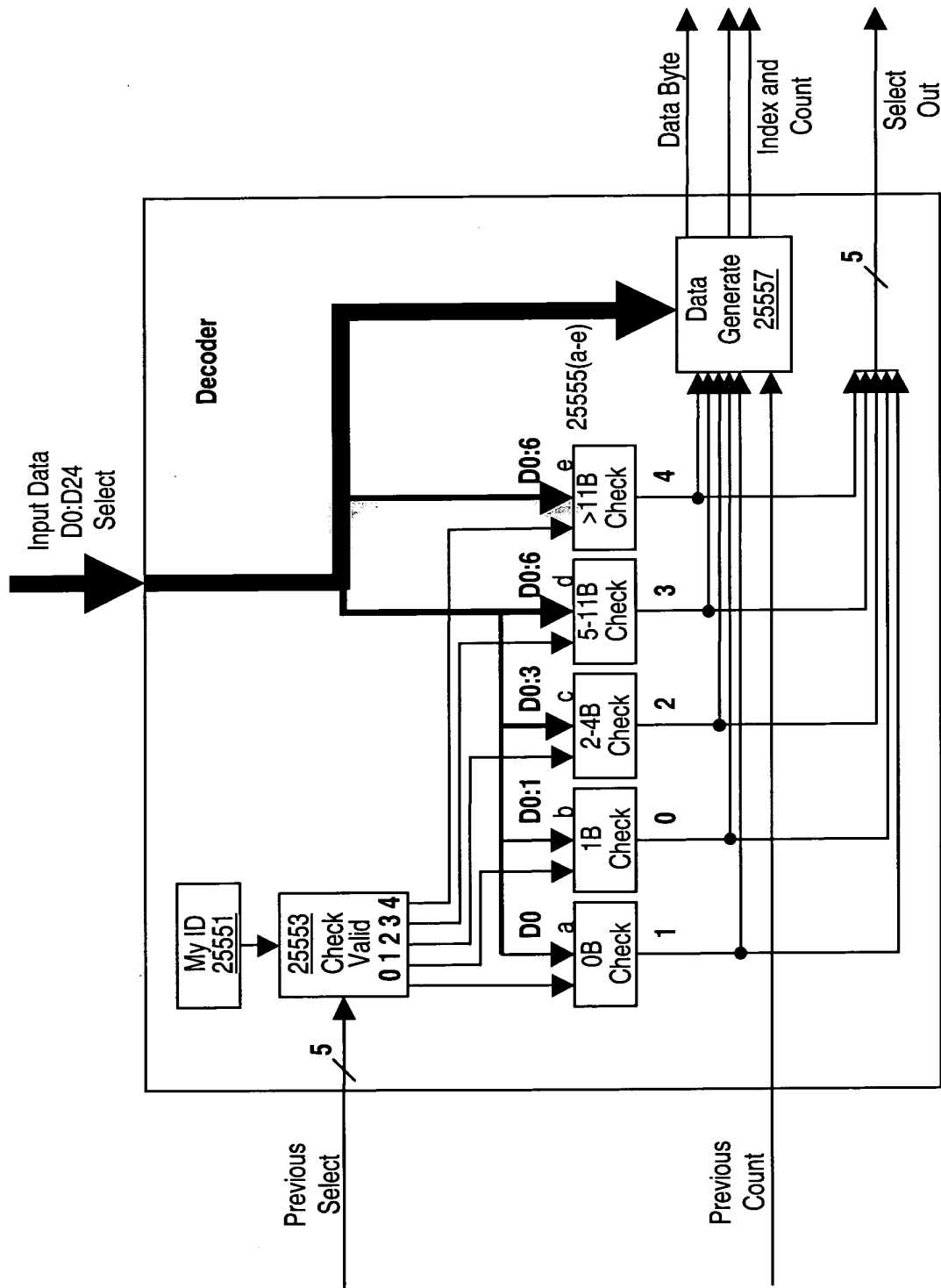


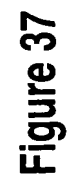
Figure 35

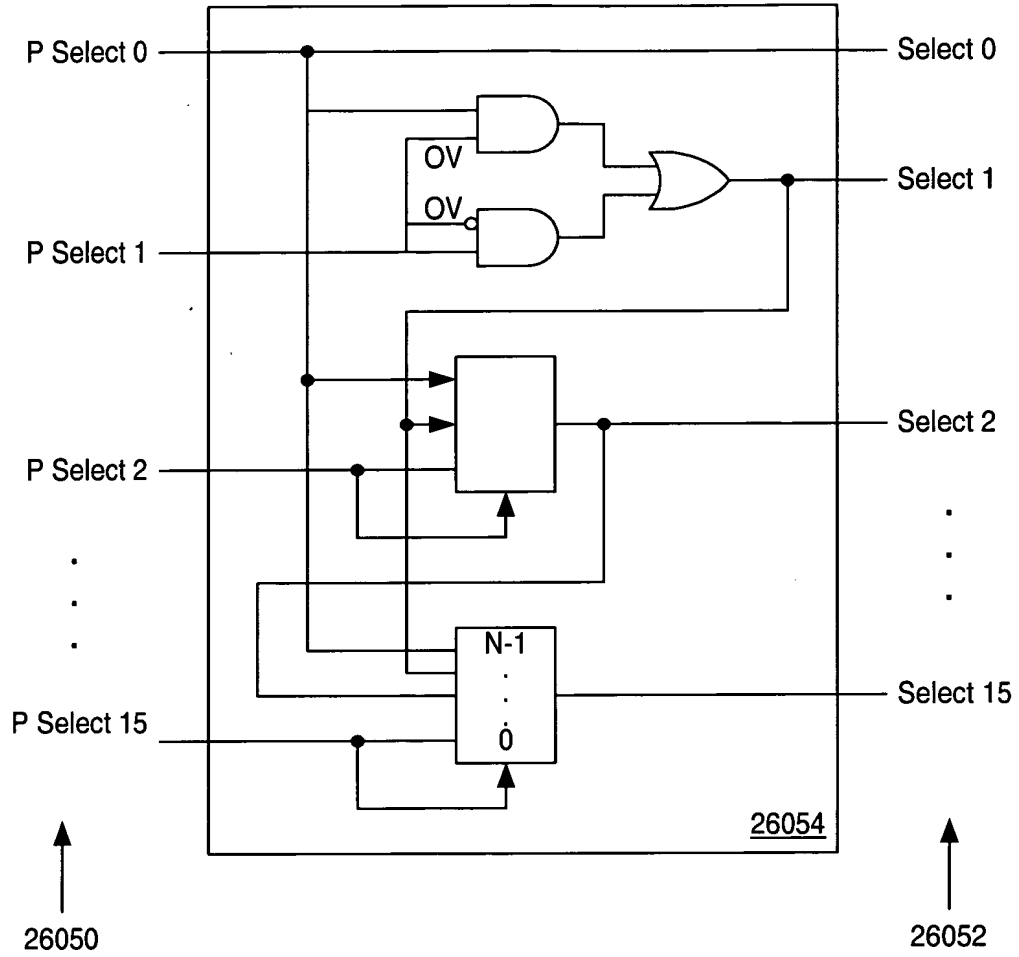
Previous Select	10	08	04	02	01	00
My ID=01	1F	1F	1F	1F	1F	00
My ID=02	1F	1F	1F	1F	1F	00
My ID=04	1F	1F	1F	1F	1F	00
My ID=08	1F	1F	1F	1F	1E	00
My ID=10	1F	1F	1F	1F	1E	00
My ID=20	1E	1E	1E	1E	00	00
My ID=40	1E	1E	1E	1C	00	00
My ID=80	08	00	00	00	00	00

Figure 36a

Select	10	08	04	02	01	00
Data Byte	X	D1:D8	X	X	X	X
Index	D2:D7	X	D4:D9	D7:D12	D7:D12	X
Count	PC+1	PC+1	D2:D3+PC+2	D4:D6+PC+5	D13:D24+PC	X

Figure 36b





Stage 3

Figure 38

004720" 0849T 960

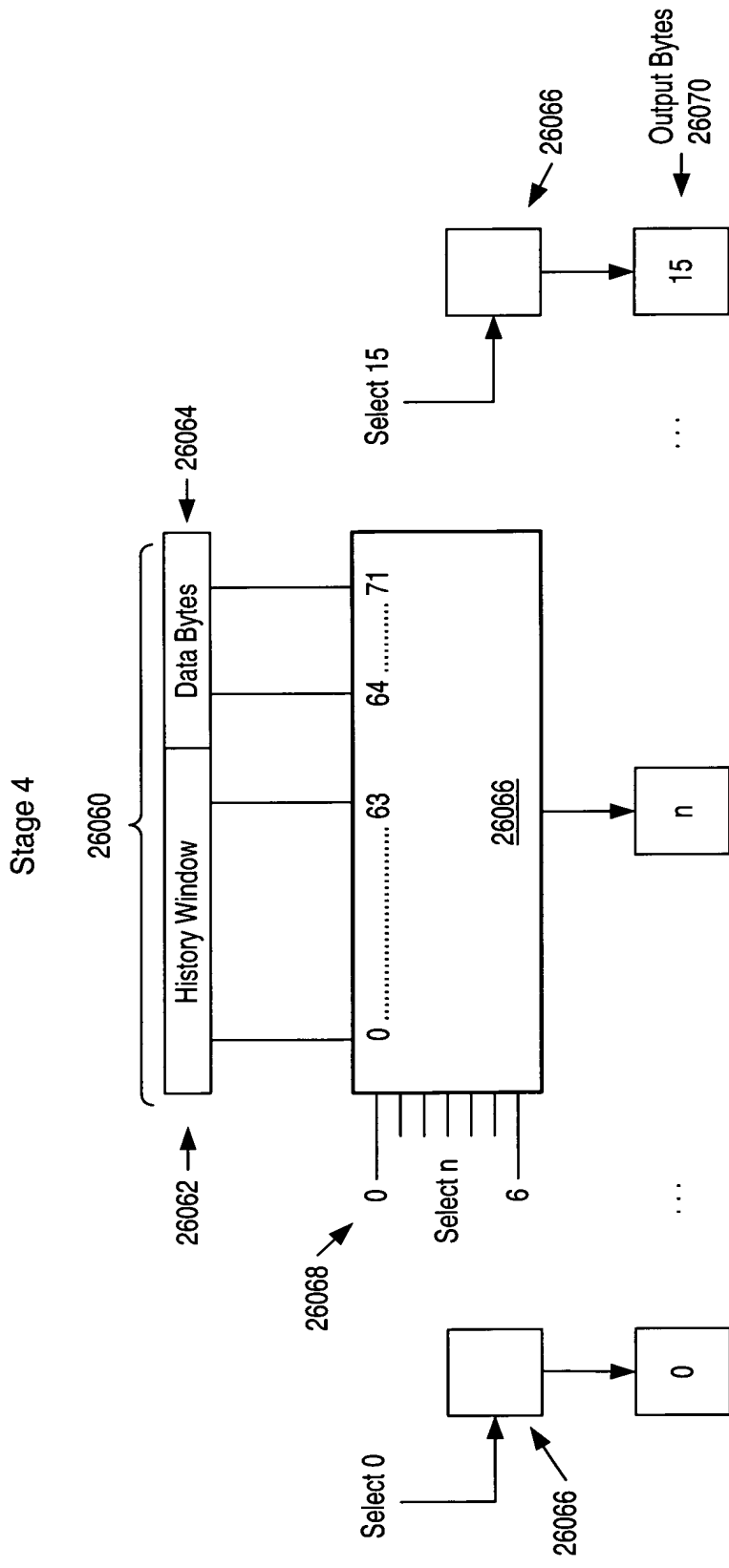
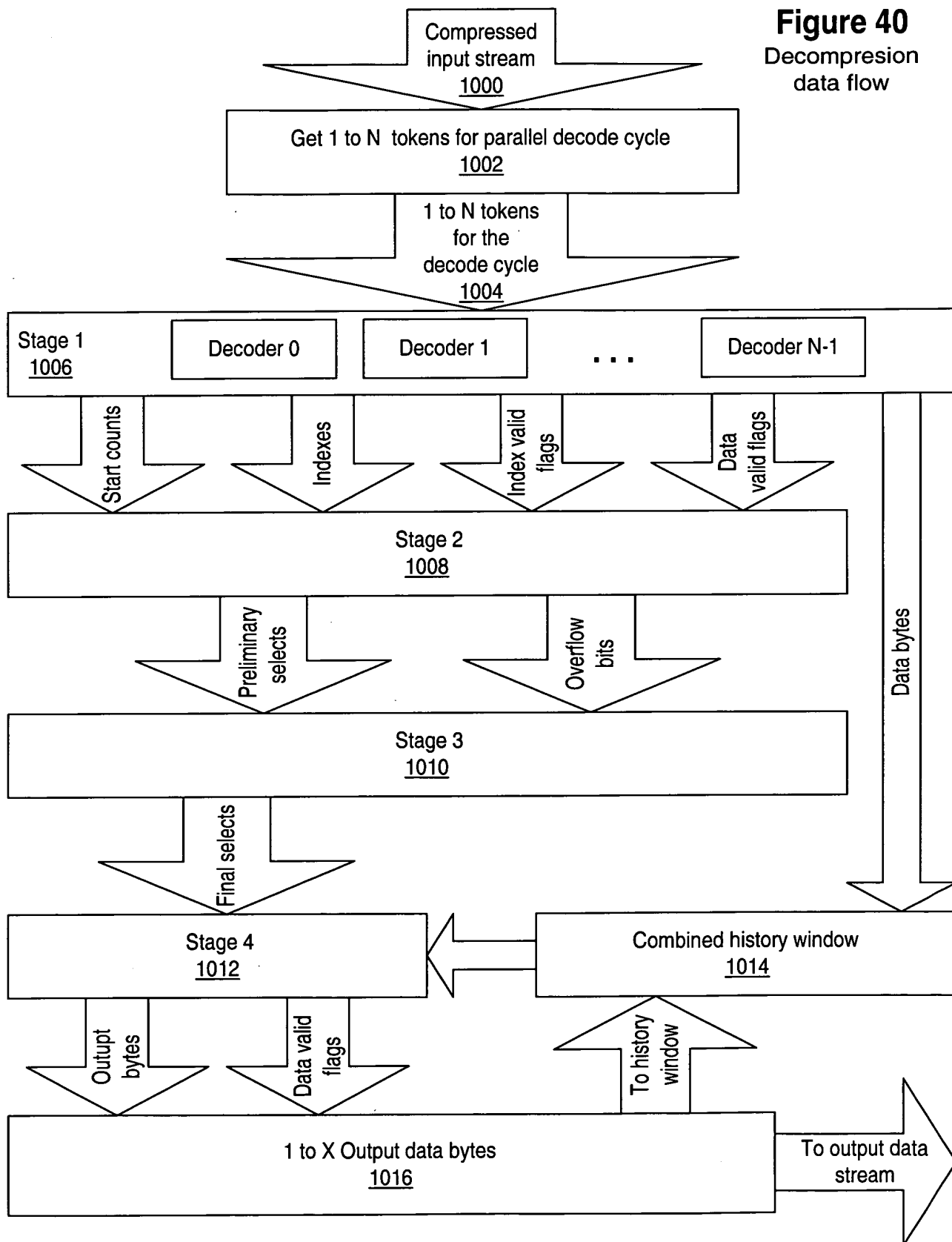
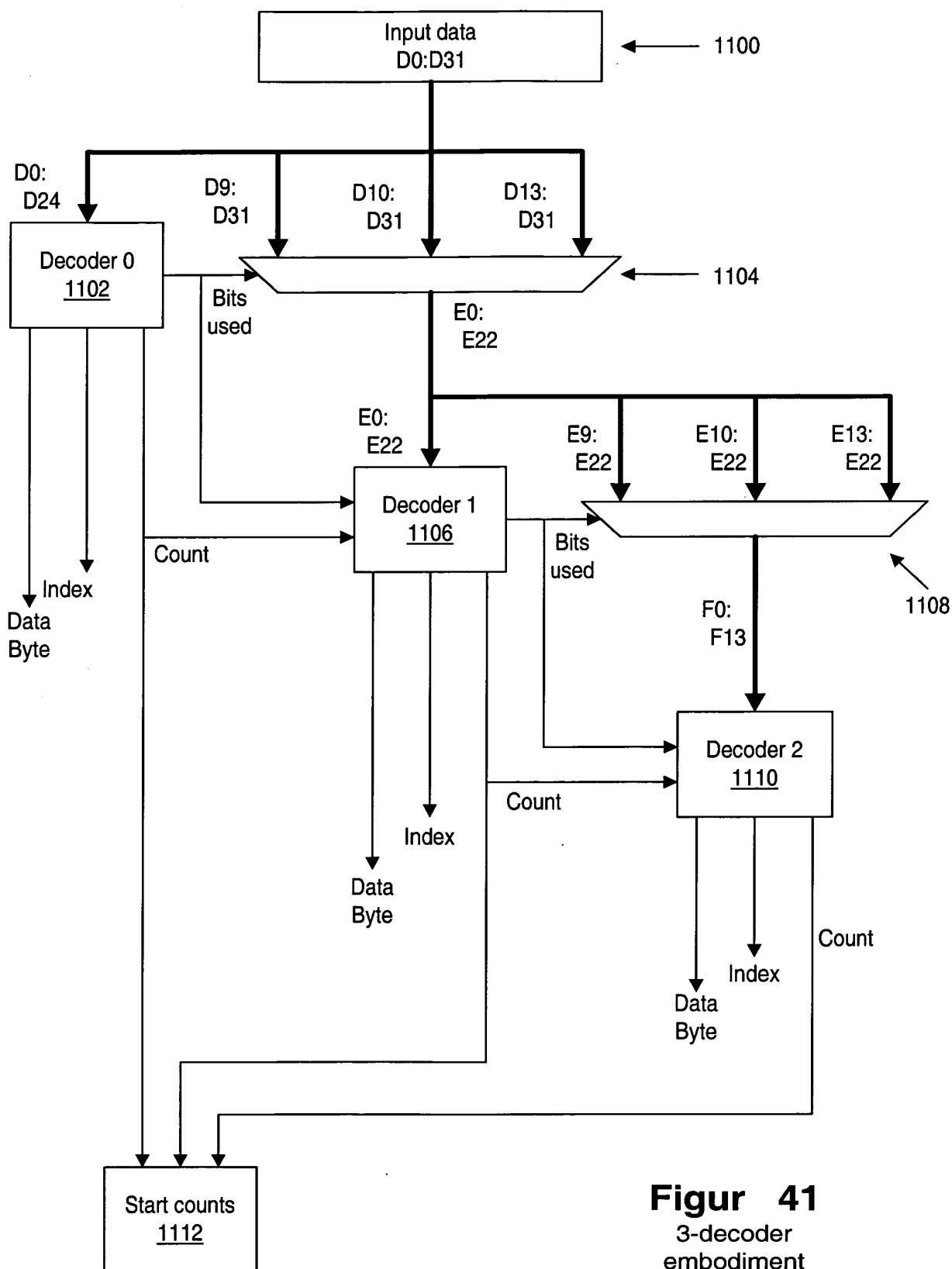


Figure 39

Figure 40
Decompression
data flow





Figur 41
3-decoder
embodiment

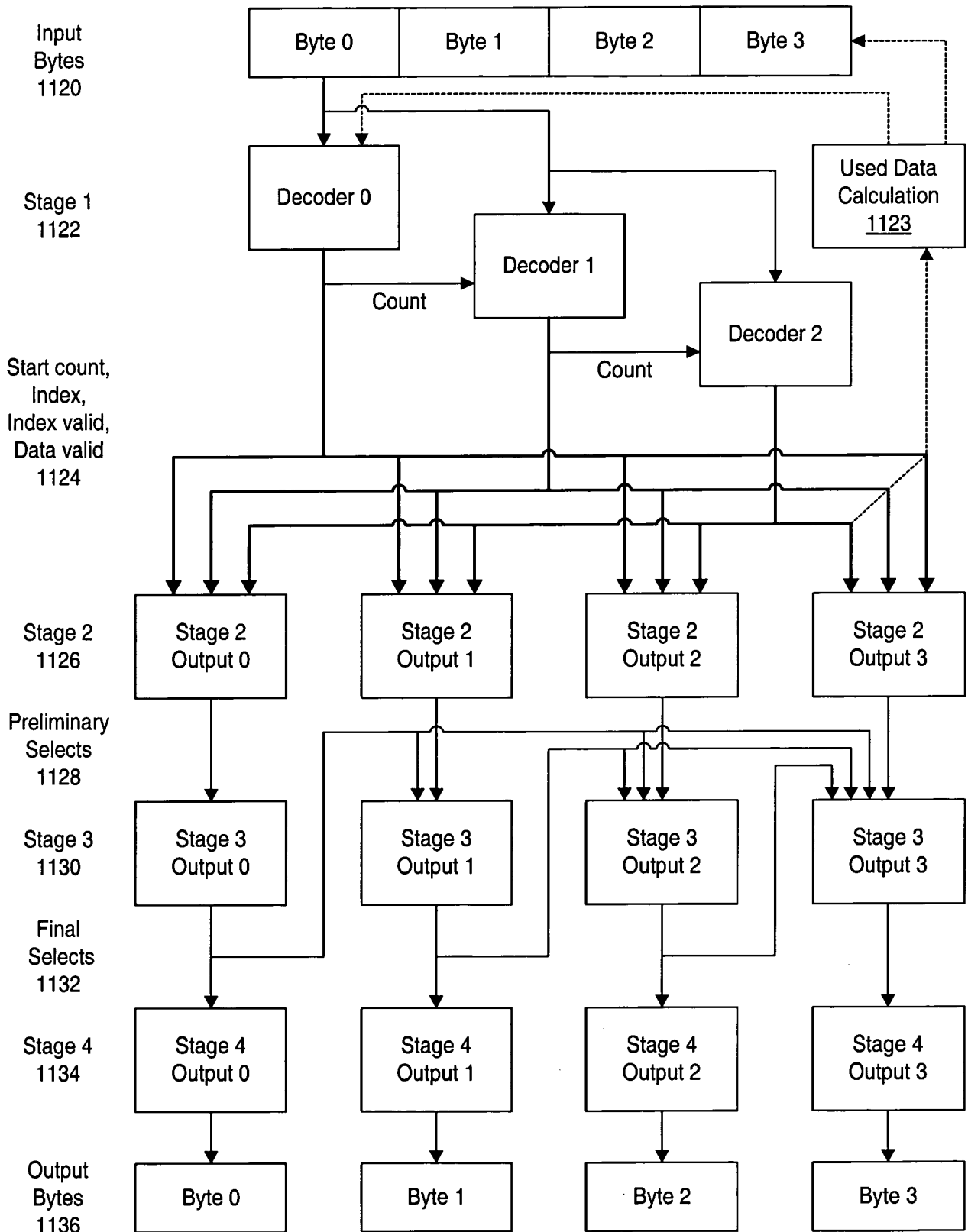


Figure 42a

- 4 input bytes, 3 decoders, 4 output bytes

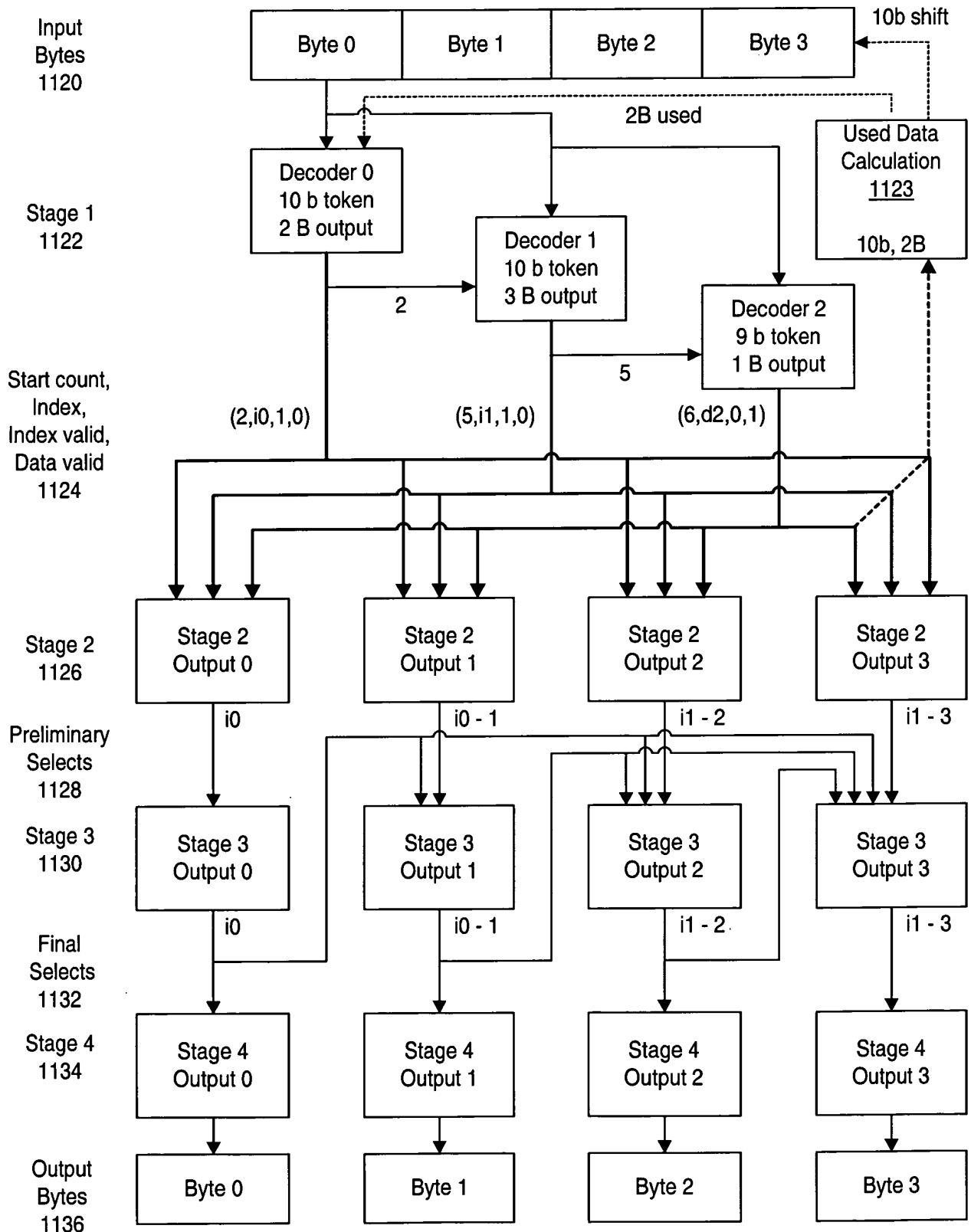


Figure 42b

- example using 4 input bytes, 3 decoders, 4 output bytes

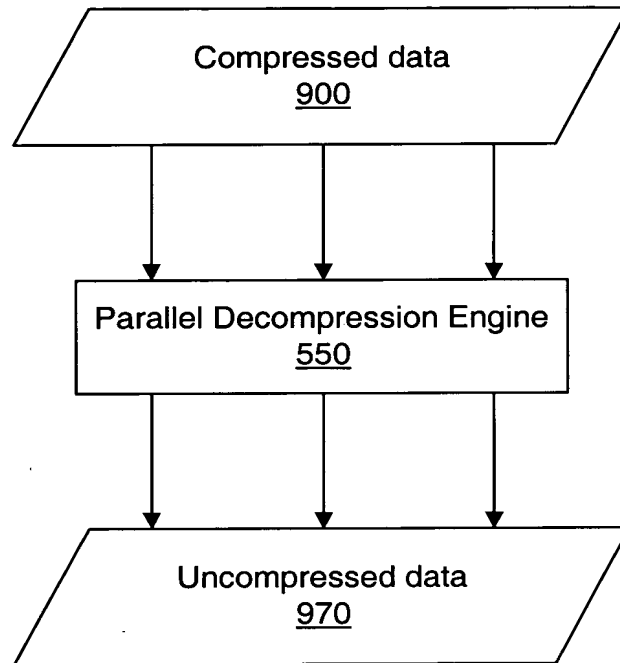


Figure 43a
Decompression
flowchart

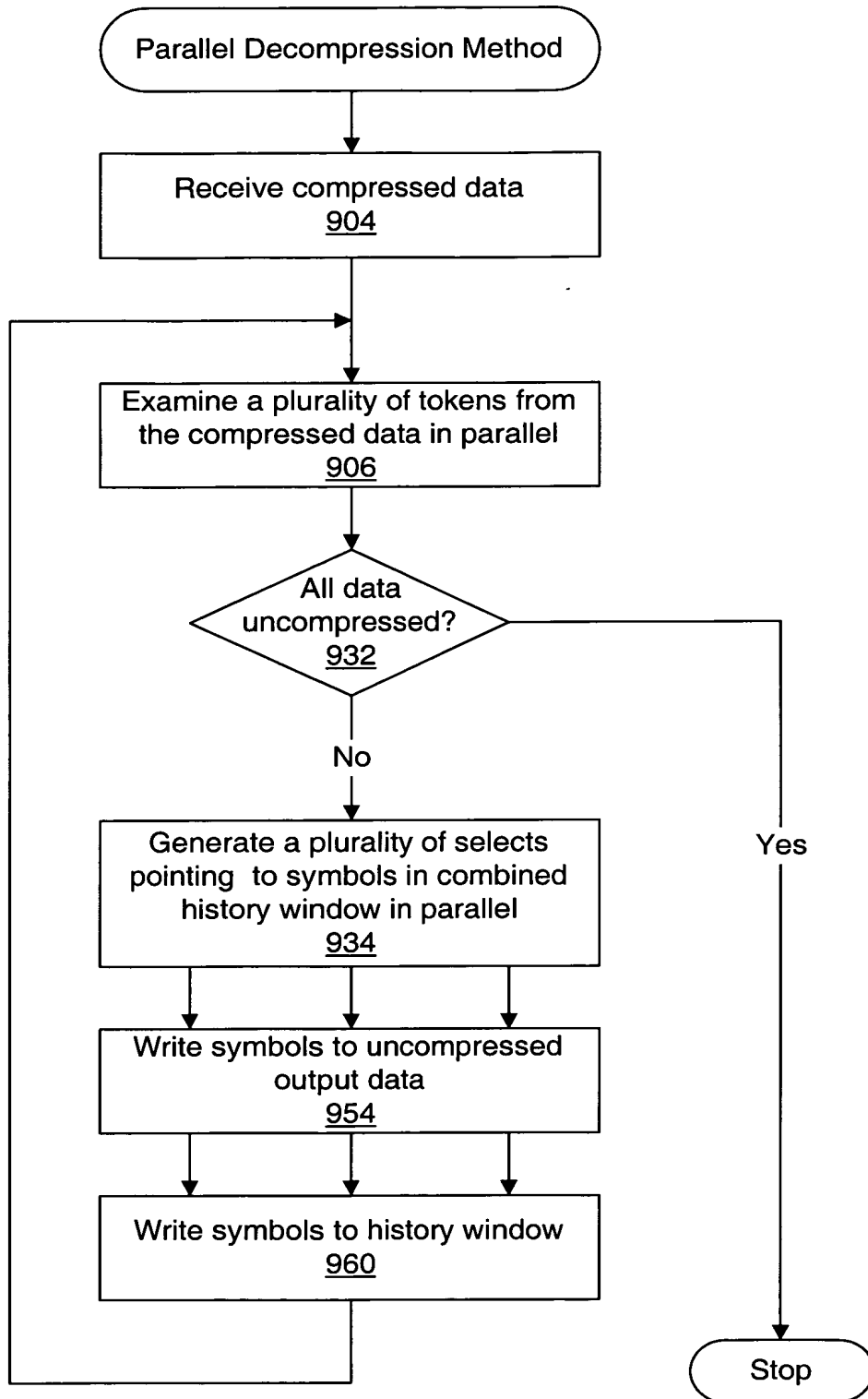


Figure 43b
Decompression flowchart

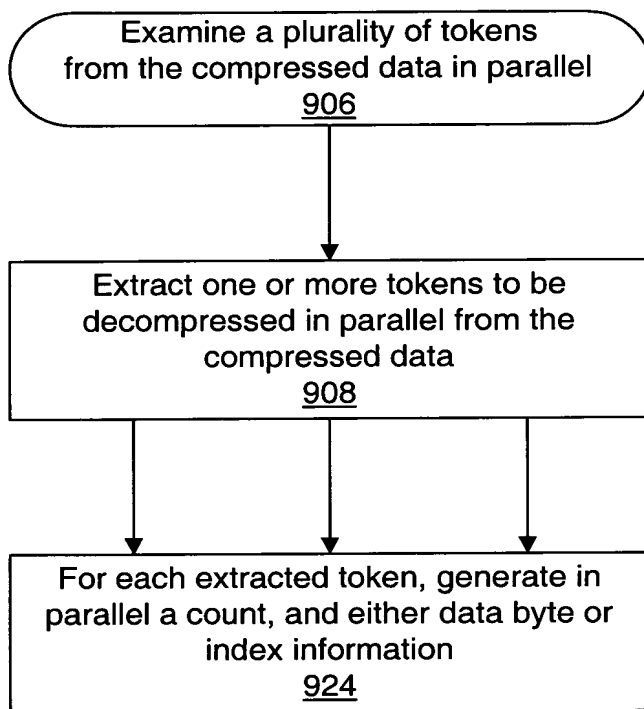


Figure 43c
Decompression flowchart


```

graph TD
    908([Extract one or more tokens to be decompressed in parallel from the compressed data  
908]) --> 910{More input data?  
910}
    910 -- No --> 922{Any valid decodes?  
922}
    910 -- Yes --> 912{Decoder available?  
912}
    912 -- No --> 922
    912 -- Yes --> 914[Determine size of token  
914]
    914 --> 915{Full token?  
915}
    915 -- No --> 922
    915 -- Yes --> 916[Determine number of bytes generated by token  
916]
    916 --> 918[Shift input data  
918]
    918 --> 920{Output width met or exceeded?  
920}
    920 -- Yes --> 924([Continue decode cycle in  
924])
    920 -- No --> 910
    922 -- Yes --> 924
    922 -- No --> Stop([Stop])
  
```

Figure 43d
Decompression flowchart

Figure 43d
Decompression
flowchart

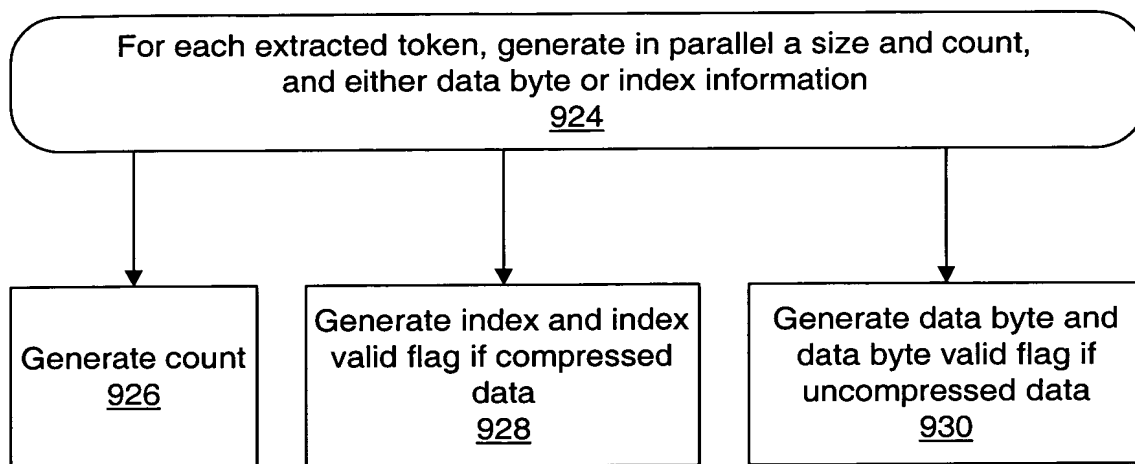


Figure 43e
Decompression flowchart

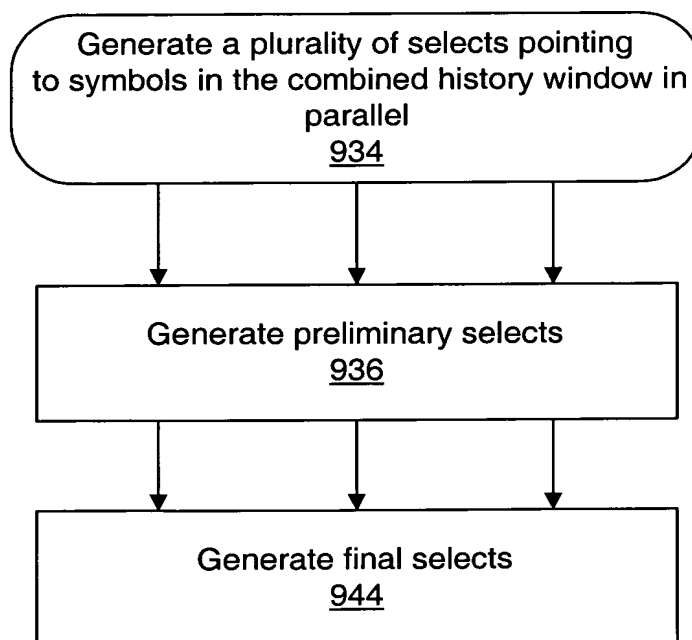


Figure 43f
Decompression flowchart

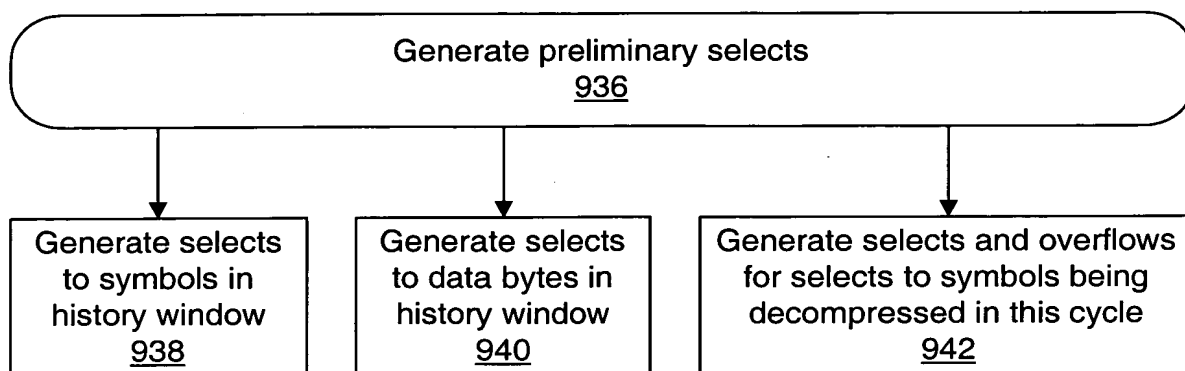


Figure 43g

Decompression
flowchart

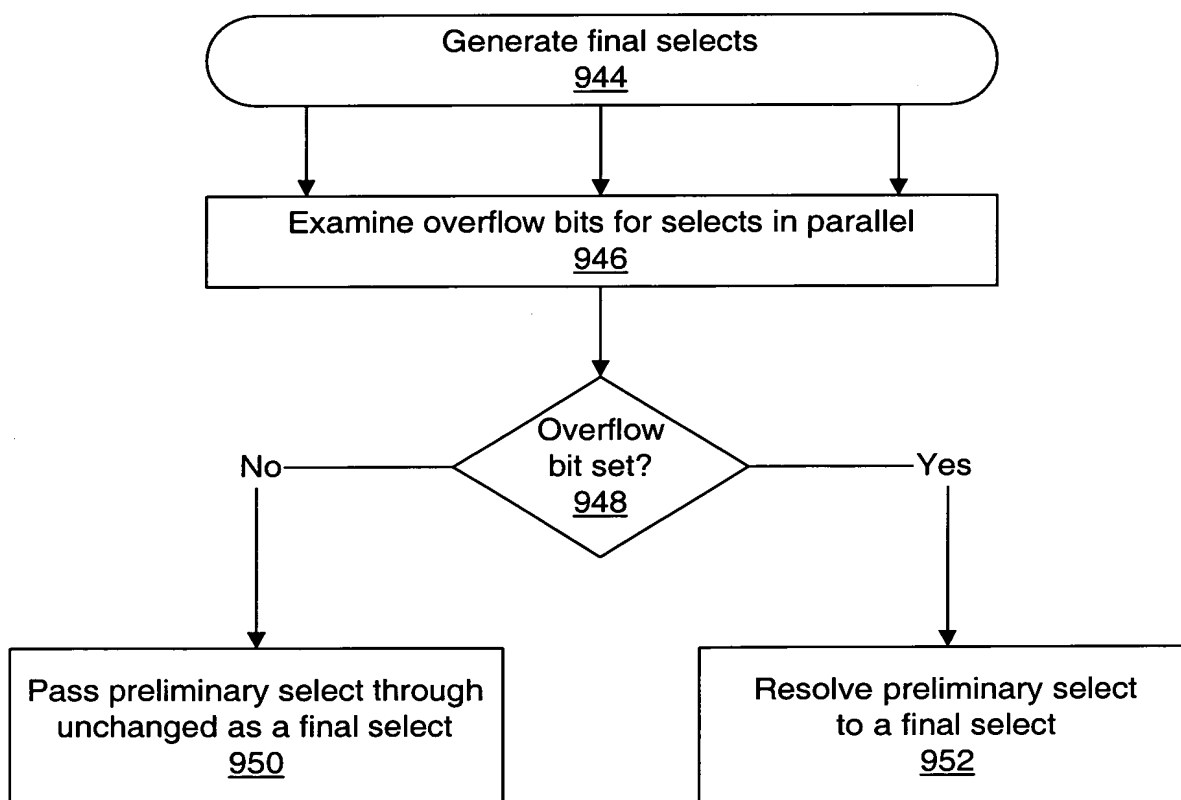


Figure 43h

Decompression
flowchart

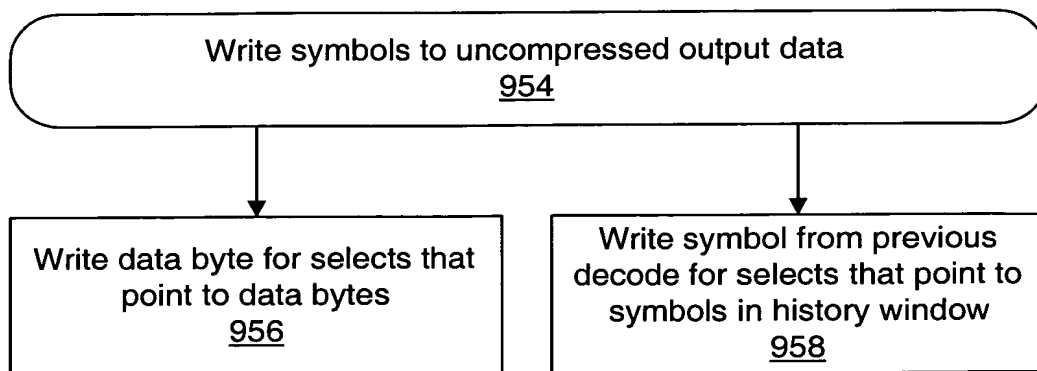


Figure 43i
Decompression flowchart

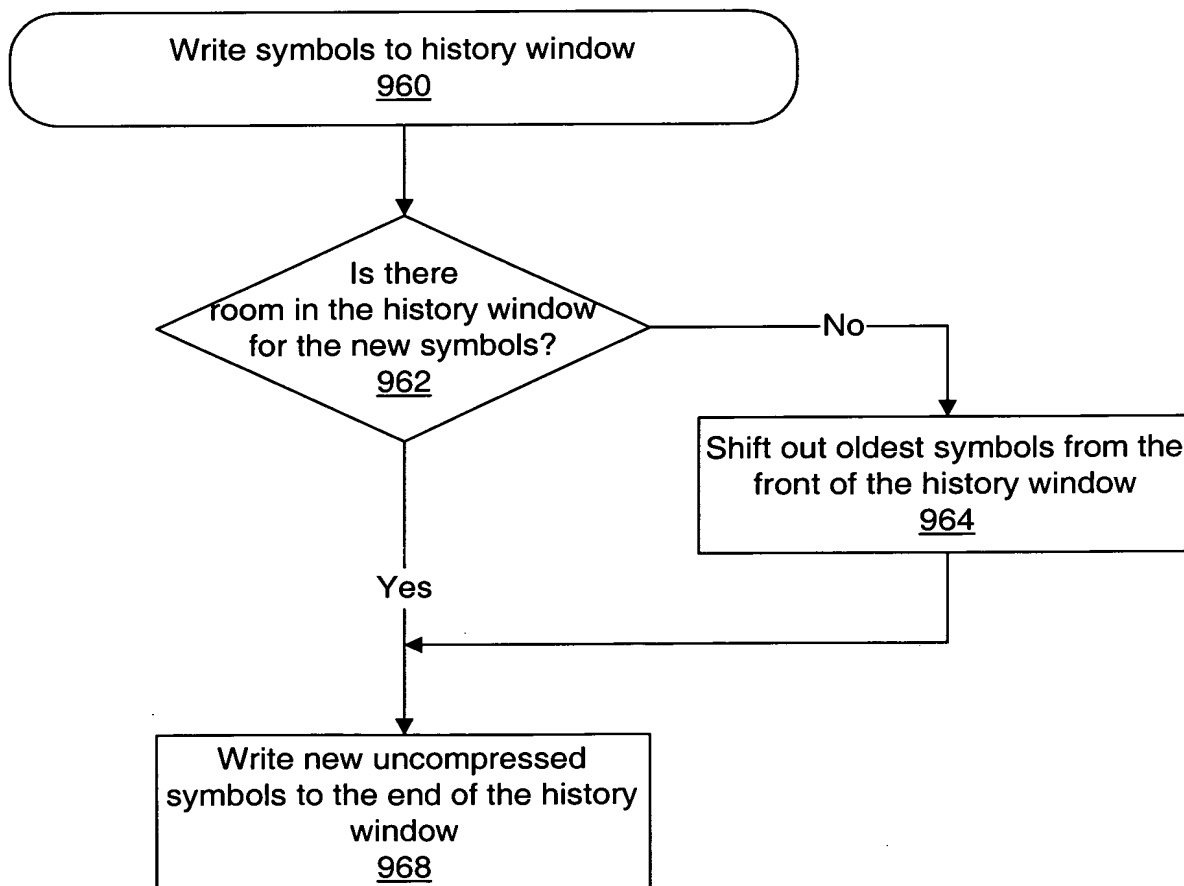


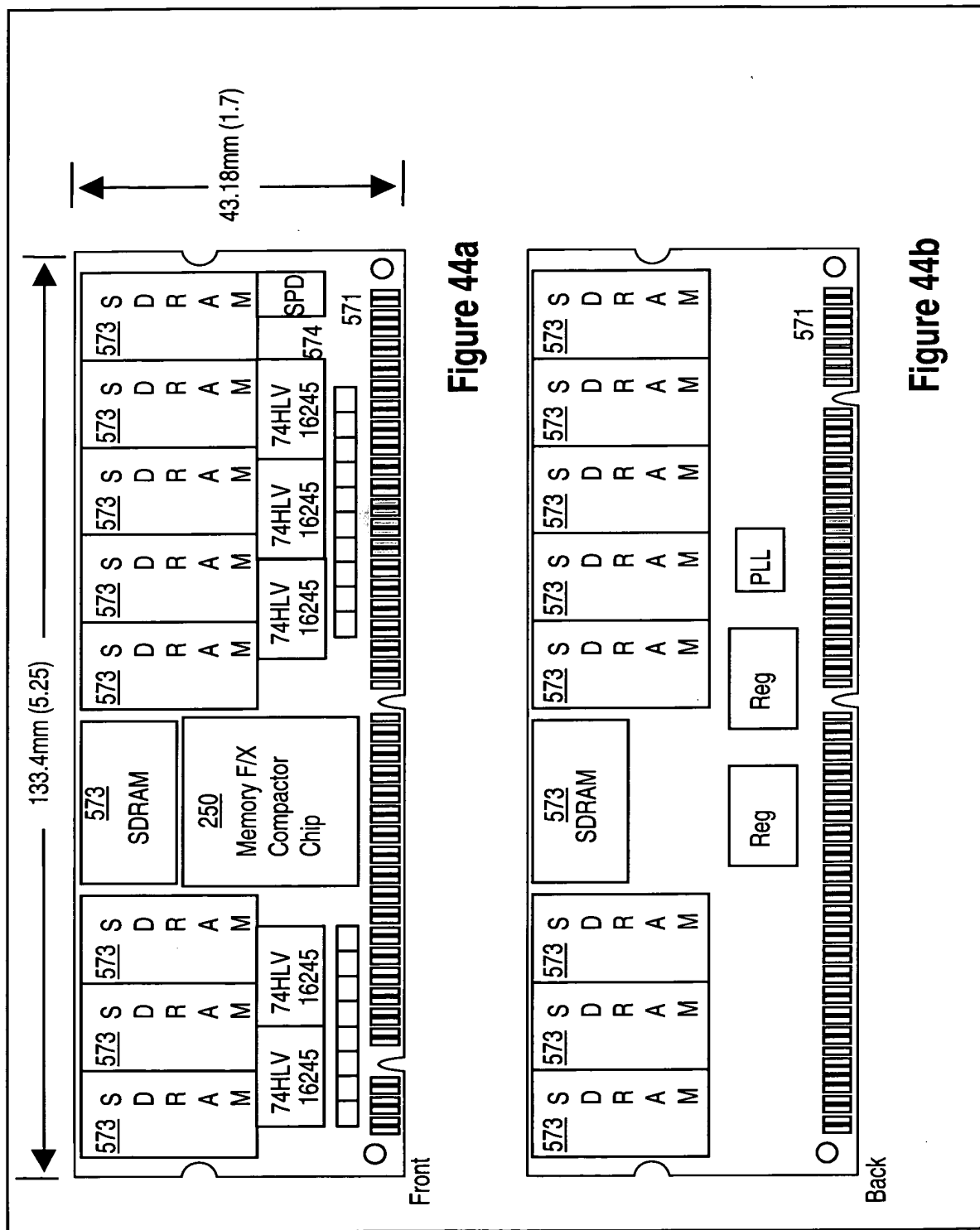
Figure 43j
Decompression flowchart

```

graph TD
    Start([Start]) --> 910{More input data?  
910}
    910 -- No --> 922{Any valid decodes?  
922}
    910 -- Yes --> 912{Decoder available?  
912}
    912 -- No --> 922
    912 -- Yes --> 914[Determine size of token  
914]
    914 --> 915{Full token?  
915}
    915 -- No --> 924[For each extracted token, generate in parallel a count, and either data byte or index information  
924]
    915 -- Yes --> 916[Determine number of bytes generated by token  
916]
    916 --> 918[Shift input data  
918]
    918 --> 920{Output width met or exceeded?  
920}
    920 -- No --> 910
    920 -- Yes --> 924
    922 -- No --> Stop([Stop])
    922 -- Yes --> 924
    924 --> 934[Generate a plurality of selects pointing to symbols in combined history window in parallel  
934]
    934 --> 954[Write symbols to uncompressed output data  
954]
    954 --> 960[Write symbols to history window  
960]
    960 --> 924

```

Decompression flowchart



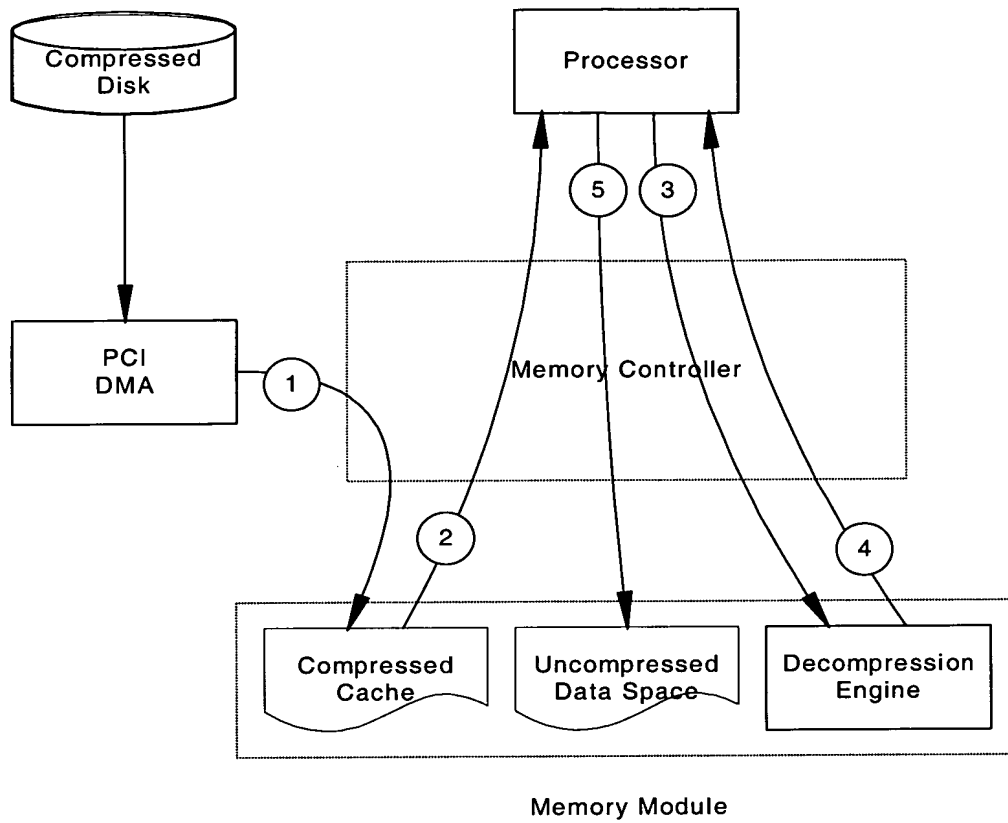


Fig. 45

The diagram illustrates the architecture of a database system, organized into four horizontal layers separated by dashed lines:

- Application Layer:**
 - Application Program:** Interacts with the Database Middleware and Application Data Space.
 - Application Data Space:** A storage area for application data, connected to the Application Program and Database Middleware.
- Database Software Layer:**
 - Database Middleware:** Acts as a bridge between the Application Program and the Database Engine.
 - Database Engine:** The core of the database, interacting with the Buffer Cache Manager and Database Data Space.
 - Buffer Cache Manager:** Manages the Buffer Cache, handling **(MemCpy Transfers)** between the Buffer Cache and the Buffer Cache Manager.
 - Buffer Cache:** A storage area for cached data, connected to the Buffer Cache Manager and File Manager.
- Operating System Layer:**
 - File Manager:** Manages files on the disk, interacting with the Buffer Cache and Raw I/O.
 - Raw I/O:** Provides raw input/output to the Device Drivers.
 - Device Drivers:** Interface between the Raw I/O and the Disk Controller.
- I/O Subsystem:**
 - Disk Controller:** Manages data flow between the Device Drivers and the Database Disk Partition, handling **(DMA Transfers)**.
 - Database Disk Partition:** The physical storage area for the database.

Arrows indicate the direction of data flow and control between components across the layers.

Fig. 46 Typical Database System

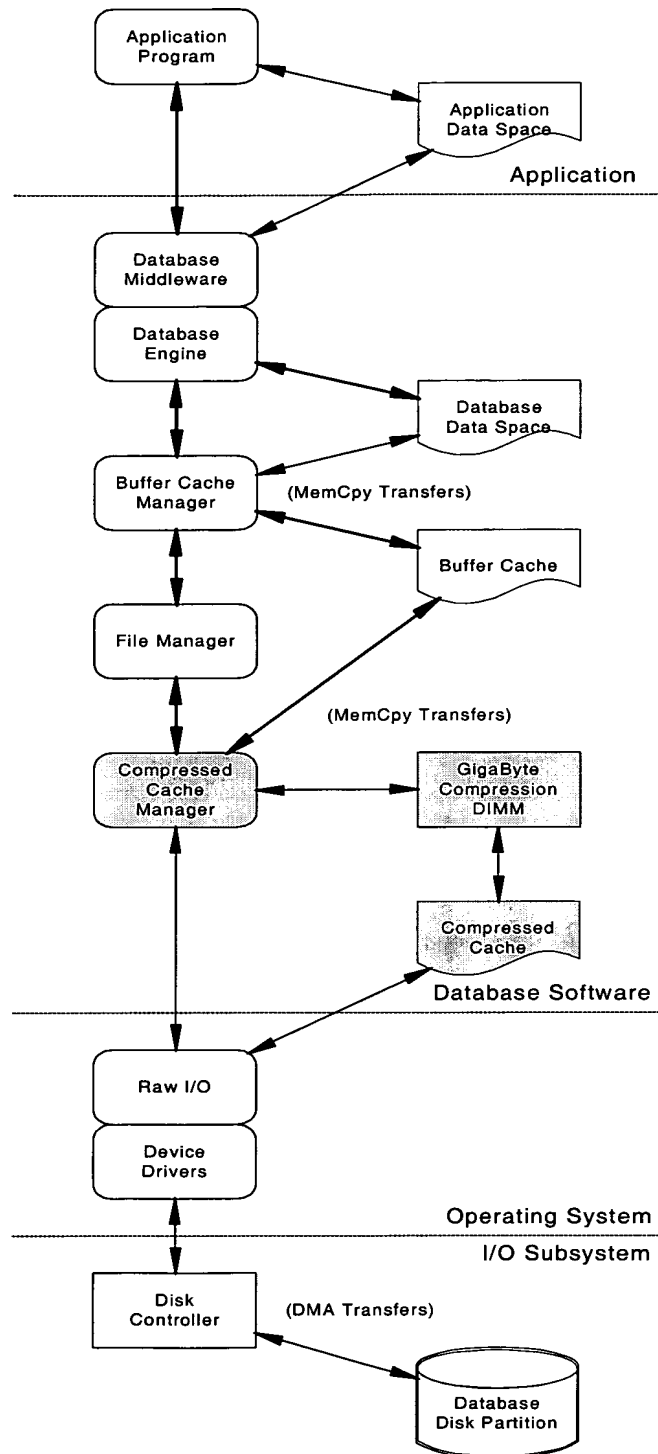


Fig 47 File System Filter Example

09616480 "071400

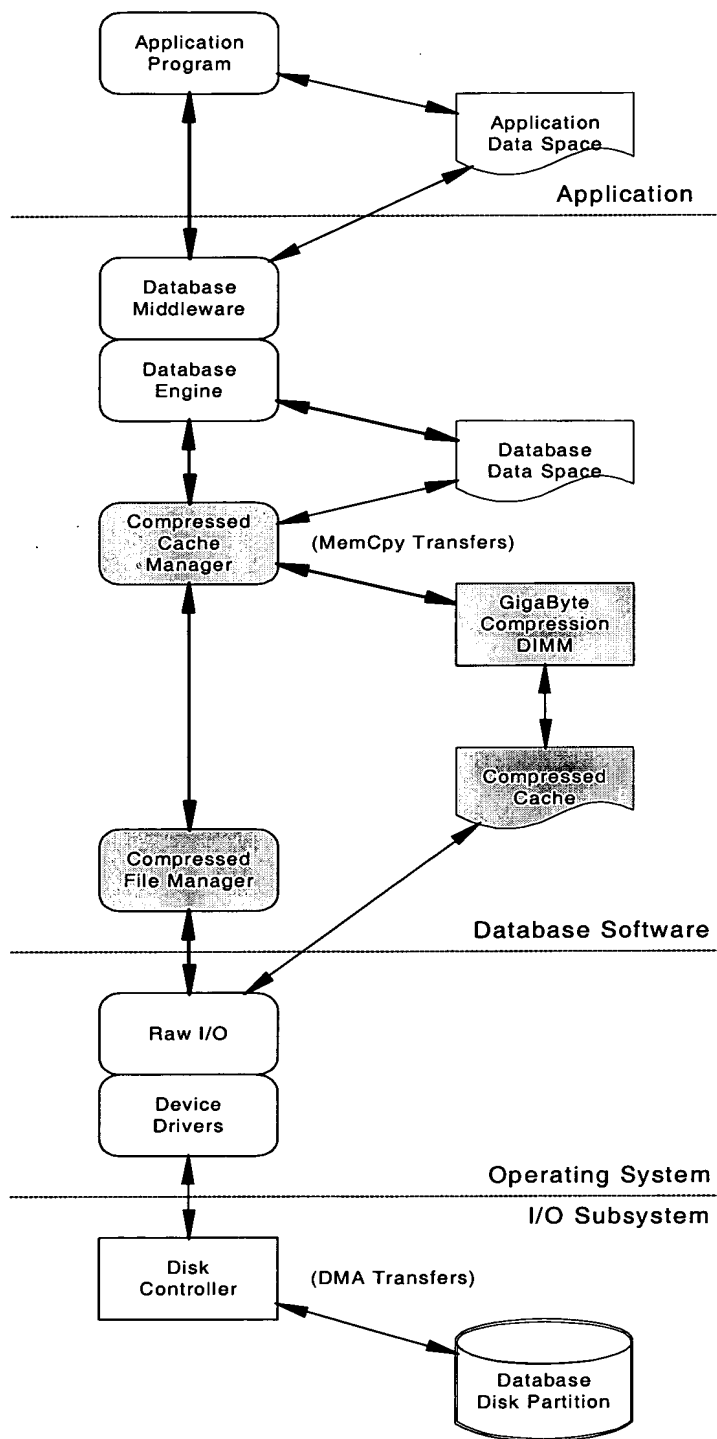


Fig. 48 Compressed Buffer Cache Example